IBM

Communications Server for z/OS V1R2 TCP/IP Implementation Guide

Volume 6: Policy and Network Management

Design and implement z/OS policies in Policy Agent using zIDS and zQoS

Covers Sysplex Distributor, IDS, and QoS polices and samples

Details SNMP and strategies for network management

Adolfo Rodriguez
Octavio Ferreira
Peter Focas
Garth Madella
Barry Mosakowski
Alan Nichols
Steve Zammit

Redbooks

ibm.com/redbooks

International Technical Support Organization

Communications Server for z/OS V1R2 TCP/IP Implementation Guide Volume 6: Policy and Network Management

October 2002

SG24-6839-00

Take Note! Before using this information and the product it supports, be sure to read the general information in “Notices” on page vii.

First Edition (October 2002)

This edition applies to Volume 1 Release 2 of Communications Server for z/OS IP.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HZ8 Building 662
P.O. Box 12195
Research Triangle Park, NC 27709-2195

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:
This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

DFS™
DPI®
IBM®
IBM eServer™
MVS™
NetView®
OS/390®
Perform™
RACF®
Redbooks™
Redbooks(logo)™
SAA®
S/390®
SecureWay®
SP™
SP1®
System/390®
TCS®
Tivoli®
VTAM®
z/OS™
zSeries™

The following terms are trademarks of International Business Machines Corporation and Lotus Development Corporation in the United States, other countries, or both:

Lotus®
Notes®
Word Pro®

The following terms are trademarks of other companies:

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

C-bus is a trademark of Corollary, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

Other company, product, and service names may be trademarks or service marks of others.

Preface

The Internet and enterprise-based networks have led to the rapidly increasing reliance upon TCP/IP implementations. The zSeries platform provides an environment upon which critical business applications flourish. The demands placed on these systems is ever-increasing, and such demands require a solid, scalable, highly available, and highly performing Operating System and TCP/IP component. z/OS and Communications Server for z/OS provide for such a requirement with a TCP/IP stack that is robust and rich in functionality. The *Communications Server for z/OS TCP/IP Implementation Guide* series (see below) provides a comprehensive, in-depth survey of CS for z/OS.

This volume covers the issues associated with managing the z/OS TCP/IP environment. First, we describe the SNMP support in CS for z/OS IP and how SNMP can be used in conjunction with other elements in your network. Then, we provide an in-depth look at managing, creating, and deploying policies in the z/OS Policy Agent. This includes policies with jurisdiction over Sysplex Distributor, Quality of Service (QoS), and Intrusion Detection Services (IDS).

Because of the varied scope of CS for z/OS, this volume is not intended to cover all aspects of it. The main goal of this volume is to provide an insight into the different functionality available for managing the CS for z/OS environment efficiently through the use of SNMP and Policy Agent. For more information, including applications available with CS for z/OS IP, please reference the other volumes in the series. These are:

- *Communications Server for z/OS V1R2 TCP/IP Implementation Guide Volume 1: Base and TN3270 Configuration*, SG24-5227
- *IBM Communications Server for z/OS V1R2 TCP/IP Implementation Guide Volume 2: UNIX Applications*, SG24-5228
- *OS/390 eNetwork Communications Server for V2R7 TCP/IP Implementation Guide Volume 3: MVS Applications*, SG24-5229
- *Communications Server for z/OS V1R2 TCP/IP Implementation Guide Volume 4: Connectivity and Routing*, SG24-6516 (redpiece available at `http://www.ibm.com/redbooks`)
- *Communications Server for z/OS V1R2 TCP/IP Implementation Guide Volume 5: Availability, Scalability, and Performance*, SG24-6517 (redpiece available at `http://www.ibm.com/redbooks`)
- *Communications Server for z/OS V1R2 TCP/IP Implementation Guide Volume 7: Security*, SG24-6840 (redpiece available at `http://www.ibm.com/redbooks`)

The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization, Raleigh Center.

Adolfo Rodriguez is a Senior IT Specialist at the International Technical Support Organization, Raleigh Center. He writes extensively and teaches IBM classes worldwide on all areas of TCP/IP. Before joining the ITSO, Adolfo worked in the design and development of CS for z/OS, in RTP, North Carolina. He holds a B.A. degree in Mathematics and B.S. and M.S. degrees in Computer Science from Duke University. He is currently pursuing the Ph.D. degree in Computer Science at Duke University, with a concentration on Networking Systems.

Octavio Ferreira is a Senior IT Specialist at IBM Brazil. He has 17 years of experience with IBM software support. His areas of expertise include z/OS, VM, SNA, TCP/IP, LAN and WAN. For the last eight years he has worked at the Area Program Support Group providing guidance and support to customers and designing networking solutions such as SNA to APPN migration and e-business solutions.

Peter Focas is a Network Systems Programmer in New Zealand. He has 12 years of experience in the SNA and TCP/IP networking field. His areas of expertise include the design and setup of SNA/SNI networks, configuration of secure TCP/IP servers using SSL/TLS, and digital certificate management.

Garth Madella is an Information Technology Specialist with IBM South Africa. He has 17 years of experience in the System/390 networking software field. He has worked with IBM for six years. His areas of expertise include VTAM, SNA, TCP/IP, and sysplex. He has written extensively on TCP/IP, sysplex, and Enterprise Extender issues.

Barry Mosakowski is a Software Engineer working in Raleigh, North Carolina at IBM's RTP site. He has eight years of experience in TCP/IP and SNA networking. He holds an MSE from Rensselaer Polytechnic Institute in Troy, New York. His areas of expertise include design, setup, and debugging of the Communication Server for z/OS TCP/IP to include the Telnet server, device drivers, socket applications, and Policy Agent.

Alan Nichols is an Independent Consultant living in Germany. He has 20 years of experience in MVS and z/OS and 10 years of UNIX and IP. He is currently working for last level IP/USS support in T-Systems.

Steve Zammit is an Advisory IT Specialist with the IBM Software Support Centre based in Vancouver Canada. Steve has 17 years of experience with IBM systems and networking, currently specializing in CS for z/OS TCP/IP. He holds a BSc degree in Applied Physics from Portsmouth Polytechnic, UK.

Thanks to the following people for their contributions to this project:

International Technical Support Organization, Raleigh Center
Bob Haimowitz, Jeanne Tucker, Margaret Ticknor, Tamikia Barrow, Gail Christensen, Linda Robinson

International Technical Support Organization, Austin Center
Julie Czubik

Communications Server for z/OS Development, Raleigh, NC
Jeff Haggar, Bebe Isrel, Van Zimmerman, Jerry Stevens, Tom Moore, Dinakaran Joseph, Greg Callis, Andrew Arrowwood, David Yang, Dave Herr

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners and/or customers.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

`ibm.com/redbooks/residencies.html`

Comments welcome

Your comments are important to us!

We want our Redbooks to be as helpful as possible. Send us your comments about this or other Redbooks in one of the following ways:

- Use the online **Contact us** review redbook form found at:

  `ibm.com/redbooks`

- Send your comments in an Internet note to:

  `redbook@us.ibm.com`

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HZ8 Building 662
  P.O. Box 12195
  Research Triangle Park, NC 27709-2195

Part 1

Introduction

1

What a policy is

Bandwidth, in the Internet and intranets of today, is an important subject. The amount of data that has to be transmitted through the Internet increases exponentially. Applications such as RealAudio, RealVideo, Internet Phone software, and video conference systems require a lot more bandwidth than the applications that were used in the early years of the Internet. Common Internet applications, such as telnet, FTP, and World Wide Web, cannot tolerate packet loss, but are less sensitive to variable delays. Most real-time applications, on the other hand, show just the opposite behavior, meaning they can compensate for a reasonable amount of packet loss but are usually very critical toward high variable delays.

This means that without any bandwidth control, the quality of these real-time streams depends on the only available bandwidth. Low bandwidth, or better unstable bandwidth, causes bad quality in real-time transmissions, for instance, dropouts and hangs. The quality of a transmission using real-time protocol RTP depends on the utilization of the underlying IP delivery service.

1.1 Policy-based networking

The idea behind policy-based networking is for network providers to express their networking goals in the form of policies. The policies can be thought of as the coding, or encapsulation, of the goals, or Service Level Agreements (SLAs) of the network providers. A policy system that provides policy-based networking typically provides the following components:

- Policy repository to store policies
- Policy Decision Points (PDPs) to retrieve, parse, and interpret policies
- Policy Enforcement Points (PEPs) to act on policy goals and modify network performance, security, etc., to implement the policy goals.

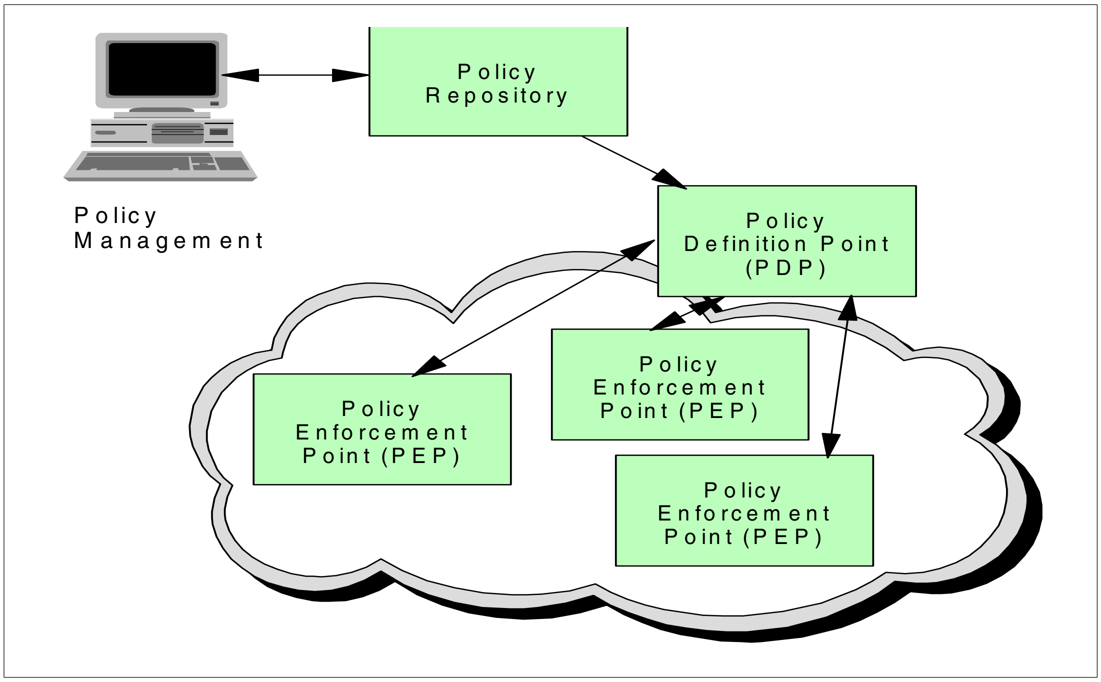

Policy Repository
Policy Management
Policy Definition Point (PDP)
Policy Enforcement Point (PEP)
Policy Enforcement Point (PEP)
Policy Enforcement Point (PEP)

Figure 1-1 Policy system model

1.2 The policy framework

The framework for defining Policies is documented in RFC 3060.

As you work in this area some terms that you will become familiar with are:

Quality of Service (QoS) Overall service (throughput, delay, etc.) received by user/application from a network

Service Policy Administrative implementation of service levels to achieve a given QoS

Integrated Services Type of service that provides end-to-end QoS using resource reservations along a network path

Differentiated Services Type of service that provides aggregate QoS to broad classes of traffic

Policy Schema Definition and syntax of object classes and attributes that make up policies on an LDAP Server

In z/OS, corporate policies are implemented through the component of the Communication Server called Policy Agent (PAGENT).

1.3 Policy model overview

Policies consist of several related objects. The main object is the *policy rule*. A policy rule object refers to one or more *policy condition*, *policy action*, or *policy time period condition objects*, and also contains information on how these objects are to be used. Policy time period objects are used to determine when a given policy rule is active. Active policy objects are related in a way that is analogous to an IF statement in a program. For example:

```
IF condition THEN action
```

In other words, when the set of conditions referred to by a policy rule are TRUE, then the policy actions associated with the policy rule are executed.

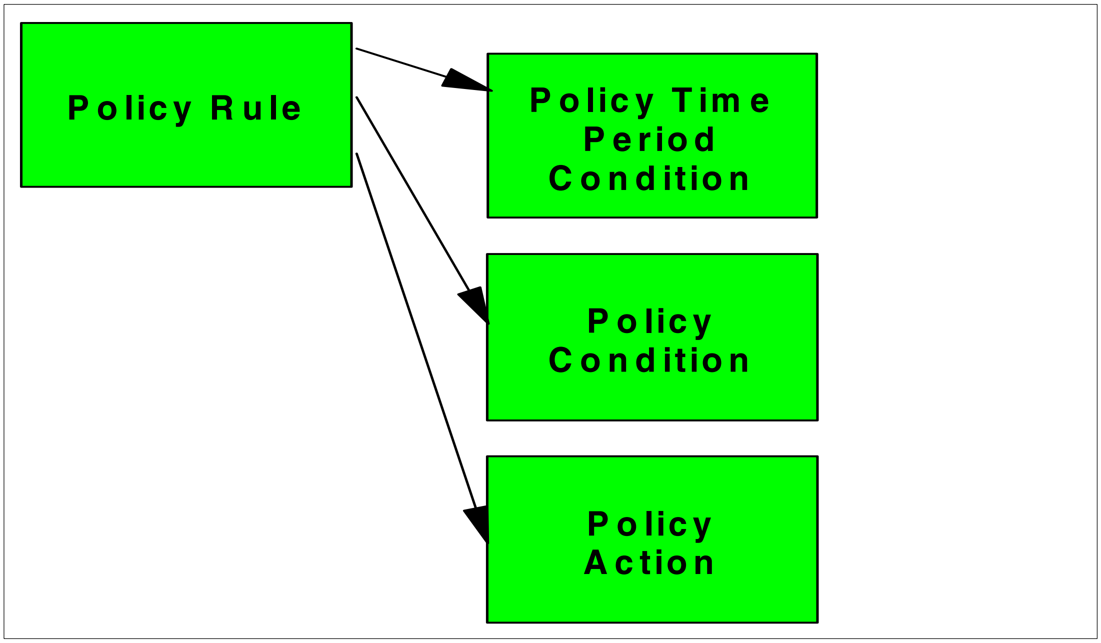


Figure 1-2 How the basic Policy objects are related

Policy rules may refer to one or more policy conditions. If a policy rule contains a single policy condition then it would be considered a simple Policy rule. Policies with multiple conditions would be considered complex Policy rules.

As per RFC 3060, there are two methods for evaluating complex Policy rules.

- Method 1: The Disjunctive Normal Form (DNF). Here we logically AND related conditions and then logically OR the result with the result of other logically ANDed related conditions.
- Method 2: The Conjunctive Normal Form (CNF). Here we logically OR related conditions and then logically AND the result with other logically ORed related conditions.

In order to accomplish these evaluations, individual policy conditions are assigned an arbitrary group number, and also an indication of whether the condition is negated. This should become clear in with the following example.

Consider two Condition Sets:

1. Conditions C1, C2, and C3 make up group number 1
2. Conditions C4 and C5 make up group number 2

Note: Condition C2 is negated.

C1: Group number = 1, Condition negated = FALSE
C2: Group number = 1, Condition negated = TRUE
C3: Group number = 1, Condition negated = FALSE
C4: Group number = 2, Condition negated = FALSE
C5: Group number = 2, Condition negated = FALSE

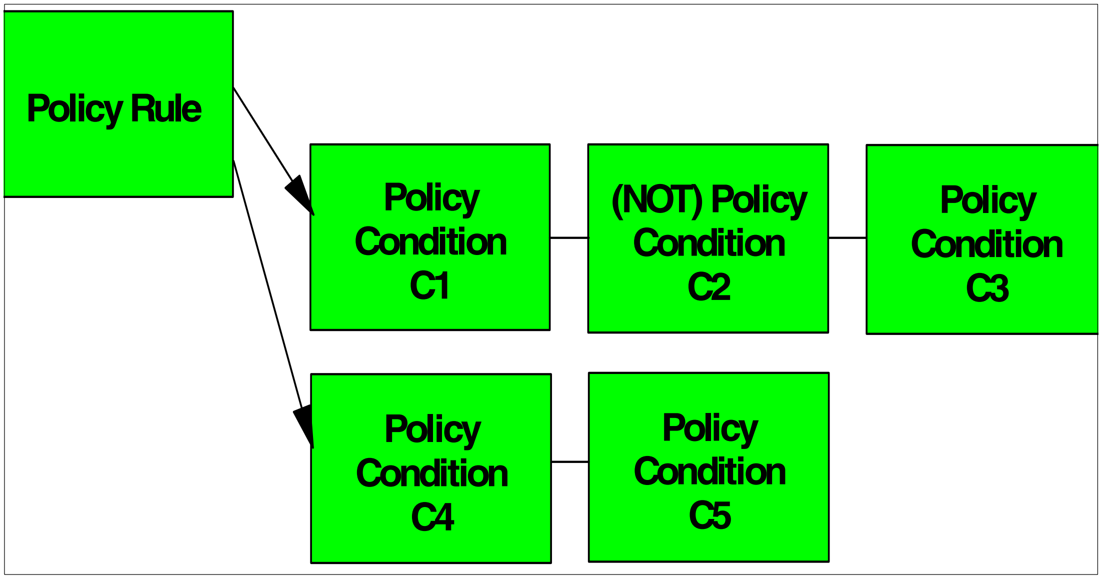


Figure 1-3 Complex policy conditions

Applying the DNF evaluation to these Condition Sets results in the policy rule matching when:

```
(C1 AND (NOT C2) AND C3) OR (C4 AND C5)
```

Applying the CNF evaluation to these Condition Sets results in the policy rule matching when:

```
(C1 OR (NOT C2) OR C3) AND (C4 OR C5)
```

Policy actions specify actions to take when the set of conditions for a policy rule's evaluations match. Although the policy model allows multiple actions for a policy rule, in practical terms a single action is all that usually makes sense for the types of policies supported by the Policy Agent.

Policy conditions and actions can either be specific to a single rule, or be reusable among several policy rules. To allow either type of conditions and actions, and to specify related information, such as condition group number and negation indicator, several other policy objects are required. First are policy condition association and policy action association objects. These objects contain condition and action related attributes, respectively, and may directly contain policy conditions and actions (rule-specific).

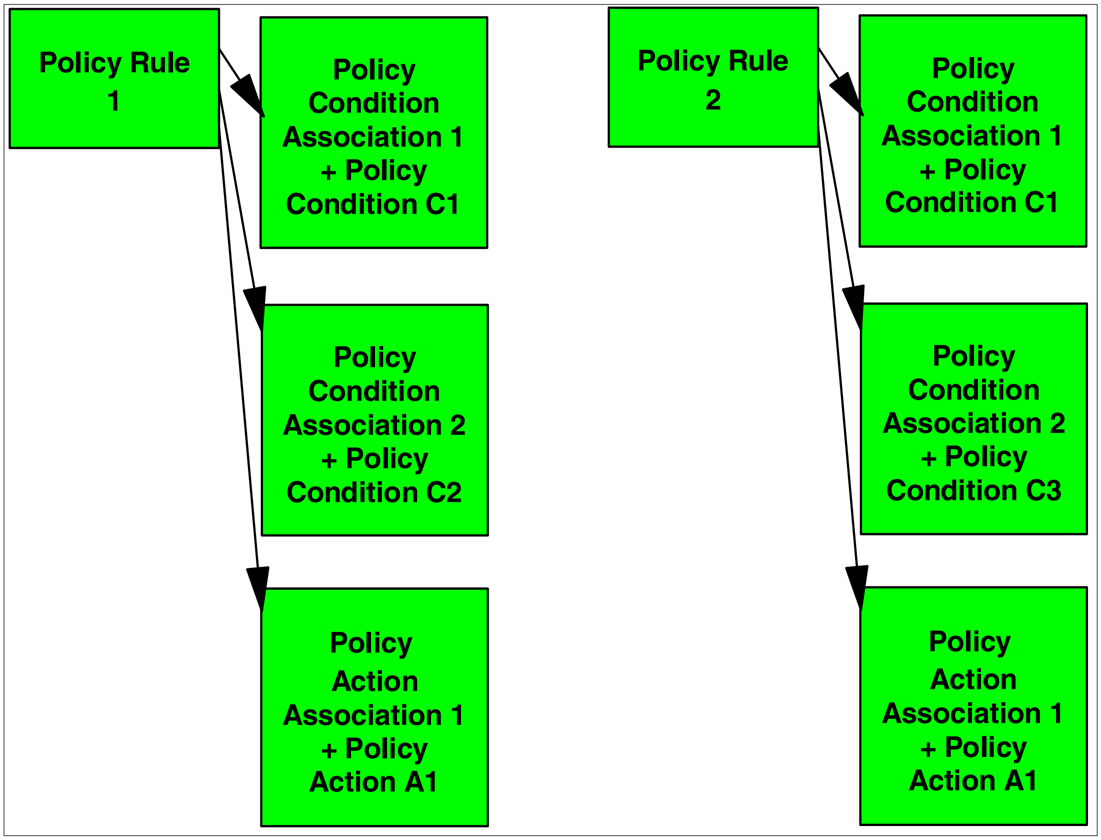

Policy Rule 1
Policy Condition Association 1 + Policy Condition C1
Policy Condition Association 2 + Policy Condition C2
Policy Action Association 1 + Policy Action A1
Policy Rule 2
Policy Condition Association 1 + Policy Condition C1
Policy Condition Association 2 + Policy Condition C3
Policy Action Association 1 + Policy Action A1

Figure 1-4 Rule-specific conditions and actions

The policy association objects alternatively may refer to conditions and actions (reusable). *Policy condition instance* and *policy action instance* objects are used to represent reusable policy conditions and actions, respectively.

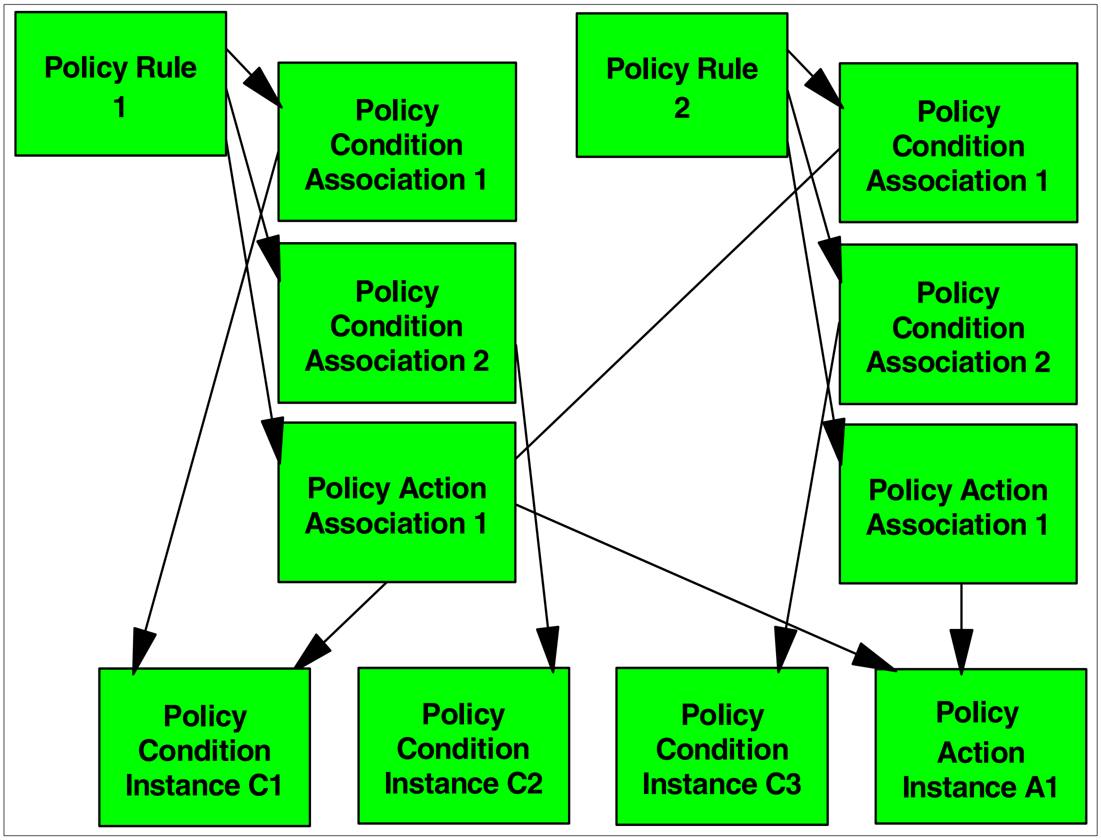

Policy Rule 1
Policy Condition Association 1
Policy Condition Association 2
Policy Action Association 1
Policy Rule 2
Policy Condition Association 1
Policy Condition Association 2
Policy Action Association 1
Policy Condition Instance C1
Policy Condition Instance C2
Policy Condition Instance C3
Policy Action Instance A1

Figure 1-5 Reusable conditions and actions

Primarily for administrative grouping of policy rules, the policy group object is used. Policy groups can refer either to policy rules or to policy groups. This allows related policy rules to be grouped together, and also allows policy groups to be grouped to any needed level of nesting.

Shown in Figure 1-6 on page 9 are two sets of rules that are related in some way, perhaps:

- Similar types of rules (QoS or IDS)
- Representing common geographic areas
- Intended to be retrieved by a given PDP

Rules 1, 2, and 3 are grouped together into group 2. Rules 4 and 5 are grouped together into group 3. Groups 2 and 3 are similarly grouped together into group 1, again due to some commonality or administrative requirement.

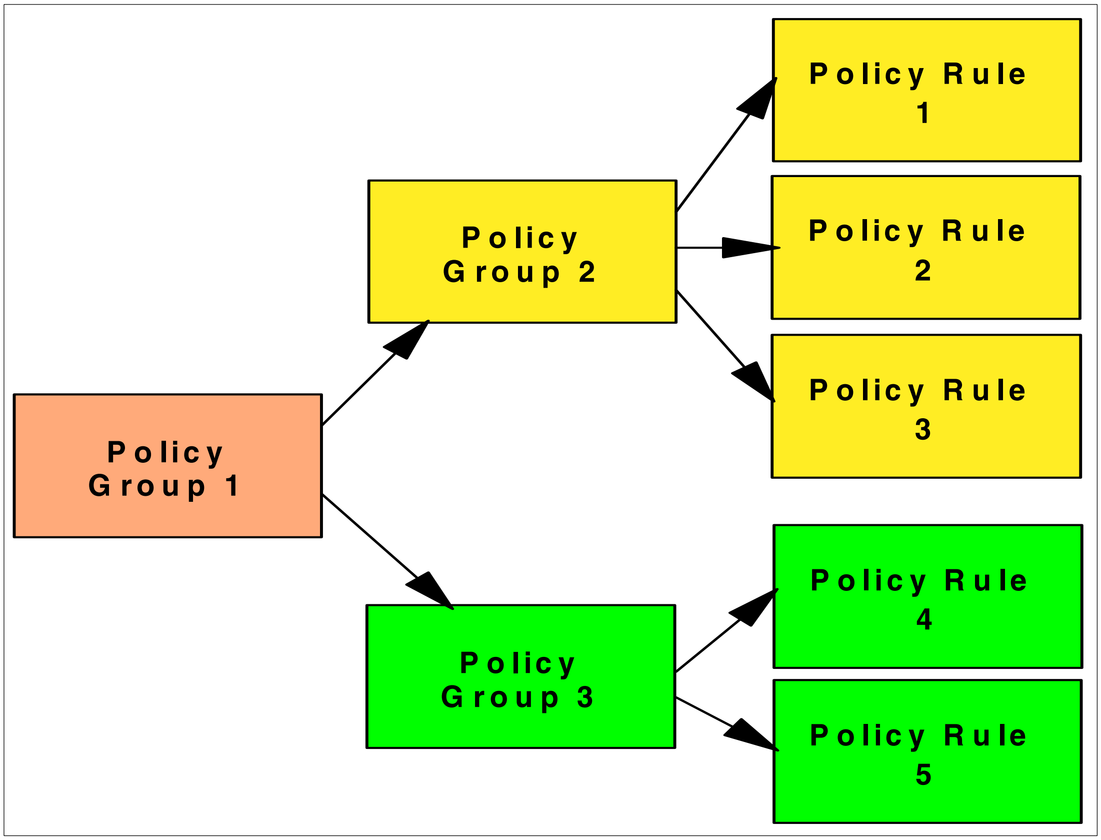


Figure 1-6 Policy groups

2

SNMP overview

The Simple Network Management Protocol (SNMP) allows for architected management type actions and information.

2.1 SNMP

The network management framework for TCP/IP-based internetworks consists of the following entities:

- SNMP managers

  An SNMP manager is an application at a network management terminal that typically issues request operations (the GET and SET requests) to an agent using the SNMP protocol. For CS for z/OS 1.2 this can be the **`osnmp`** command, the NETVIEW SNMP command, and any other manager in a TCP/IP network.

  SNMP commands such as the **`osnmp`** command and the NETVIEW SNMP command can be used to access MIB object information. They send SNMP requests to SNMP agents on either local or remote hosts. The **`osnmp`** command supports SNMPv1, SNMPv2c and SNMPv3. The NETVIEW SNMP command supports only SNMPv1.

  > **Note:** The SNMP command that can be issued in the UNIX System Services shell environment is provided. It has exactly the same function and syntax as the **`osnmp`** command.

- SNMP agents

  An SNMP agent is a server at a managed host that responds to the SNMP protocol operations. An agent must be present at each IP host in order to enable management of that host by an SNMP manager. An SNMP agent is basically a server that provides support for the Management Information Base (MIB) objects that are pertinent to the IP host on which it resides.

- SNMP subagents

  Commonly, an SNMP agent does not maintain the values of all MIB objects directly. These values are generally maintained or accessed through one or more subagents. An SNMP subagent interacts with an agent to respond to the GET and SET requests for a particular set of objects it owns.

  The SNMP agent supports the Distributed Protocol Interface (DPI), an interface by which subagents can communicate with the SNMP agent to support particular MIB objects.

  The following three SNMP subagents are shipped with CS for z/OS 1.2:

  - The TCP/IP subagent reports information about the TCP/IP stack.
  - The OMPROUTE subagent reports the information specific to OSPF.
  - The SLA Subagent reports information about defined service policies and performance statistics related to traffic using those policies.

  At ITSO Raleigh, we used the SLA Subagent to monitor the performance statistics controlled by the z/OS Service Policy agent.

See Figure 2-1 on page 13 for an overview diagram of the CS for z/OS 1.2 SNMP support.

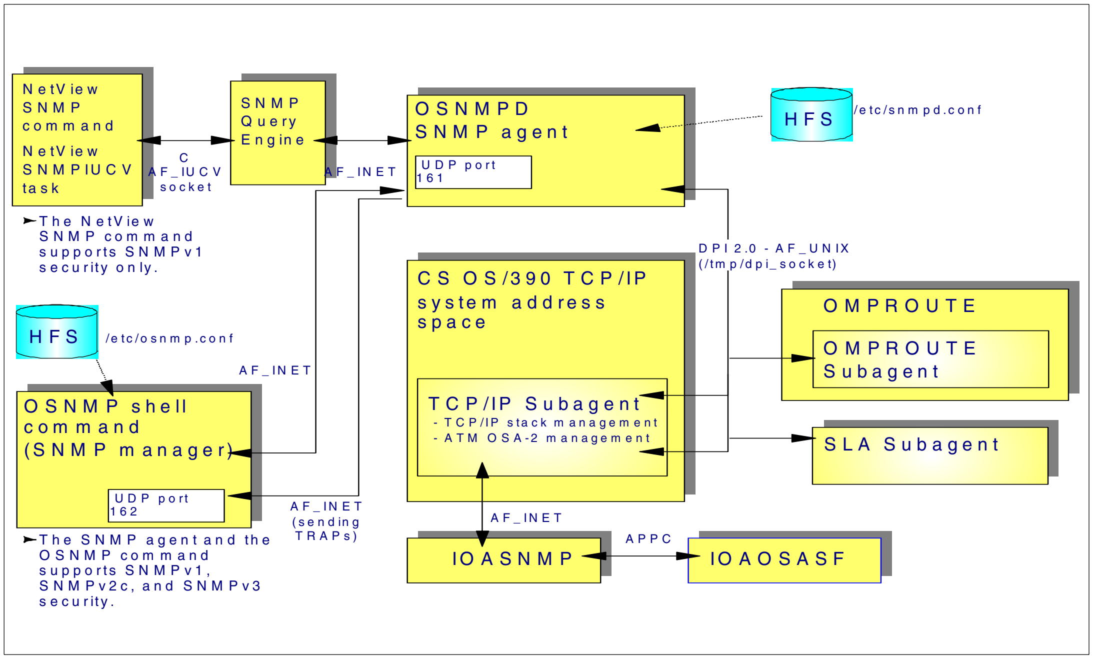


Figure 2-1 SNMP support in CS for z/OS 1.2

The SNMP manager (the **osnmp** command) and SNMP agent shipped all of the three versions of the SNMP protocol as part of CS for OS/390 V2R10 IP support.

Note: The NETVIEW SNMP command supports SNMPv1 only.

The overview of SNMP versions is shown below:

- SNMPv1

  SNMPv1 is defined in RFC 1155 and RFC 1157 and is the initial version of SNMP, which is supported by TCP/IP for MVS V3R2. SNMPv1 provides community-based security and supports the following five protocol data units (PDUs):

  - GET
  - GETNEXT
  - SET
  - RESPONSE
  - TRAP

- SNMPv2

  SNMPv2 adds additional SNMP operations such as GETBULK, provides more granular error responses, and defines an additional ability to support an enhanced MIB definition, called SMIv2.

  The SNMPv2 protocol and MIB structure are defined in RFC 1902 through RFC 1908, which in August 1993 became a proposed standard with the status of elective. There are two choices of security defined by RFCs with SNMPv2. One is named SNMPv2c, which supports community-based security like SNMPv1, and the other is SNMPv2u, which provides user-based security and allows installations to define user security names, authentication, and privacy keys. SNMPv2u was defined as an experimental protocol and became a precursor to SNMPv3. CS for OS/390 V2R10 IP supports SNMPv2c but not SNMPv2u.

Note: CS for OS/390 did support SNMPv2u in V2R5 and V2R6. Its support was dropped in V2R7 because the standards-based SNMPv3 support was introduced.

- SNMPv3

  SNMP Version 3 (SNMPv3), defined in RFCs 2570 through 2575 issued in April 1999, is the standards-based solution to the previous weaknesses of SNMP security. The SNMPv3 architecture is modularized so that portions of it can be enhanced over time without requiring the entire architecture to be replaced. SNMPv3 defines a framework that consists of (among other things):

  – A message processing model (SNMPv3)
  – A security model (user-based security)
  – An access control model (view-based access control)

  The framework is structured so that multiple models can be supported concurrently and replaced over time. For example, although there is a new message format for SNMPv3, messages created with the SNMPv1 and SNMPv2 formats can still be supported. Similarly, the user-based security model can be supported concurrently with community-based security models previously used.

  SNMPv3 provides the ability to configure the agent dynamically, from either a local or remote host, and to make changes in the configuration while the SNMP agent is running. Doing SNMP agent configuration dynamically requires a good understanding of how the SNMP SET commands can be issued to create new rows or to change or delete existing rows, and a familiarity with the SNMP engine configuration tables defined in RFCs 2570 through 2575. (For additional details, see RFCs 2570 through 2575, as well as RFCs 1901 through 1910.)

Note: In April 1999 RFCs 2271 through 2275 were made obsolete by RFCs 2571 through 2575.

The Management Information Base (MIB) defines the objects that may be managed for each layer in the TCP/IP protocol. There are two versions: MIB-I and MIB-II. MIB-I was defined in RFC 1156, and is now classified as a historic protocol with a status of not recommended.

Each managed node supports only those groups that are appropriate. For example, if there is no gateway, the EGP group need not be supported. But if a group is appropriate, all objects in that group must be supported. The list of managed objects defined has been derived from those elements considered essential. This approach of taking only the essential objects is not restrictive, since the SMI provides extensibility mechanisms such as definition of a new version of the MIB and definition of private or nonstandard objects.

In Table 2-1, you will see the summary of RFCs that define the MIB objects supported byCS for z/OS 1.2. Please refer to the RFCs listed for the complete definition.

Table 2-1 SNMP MIB support RFC summary

MIB group	RFC number
Interfaces	RFC 2233
IP	RFC 2011
IP forwarding table	RFC 1354 (Obsoleted by RFC 2096)
ICMP	RFC 2011

MIB group	RFC number
TCP	RFC 2012
UDP	RFC 2013
OSPF	RFC 1850
Ethernet-like interface	RFC 2665
IP over ATM	RFC 2320
ATM interface	RFC 1695 (Obsoleted by RFC 2515)
IBM OSA Express ATM interface	IBM TCP/IP MVS Enterprise Specific MIB
SNMP	RFC 1907, RFC 2271-RFC 2275
DPI	RFC 1592
Service level agreement management	RFC 2758 (Experimental RFC)
IBM 3172	IBM 3172 Enterprise Specific MIB
IBM Remote Ping	IBM TCP/IP MVS Enterprise Specific MIB
IBM TCP/IP MVS	IBM TCP/IP MVS Enterprise Specific MIB

Support for the RFC 2665 Ethernet-like interface MIB was added in CS for OS/390 V2R10 IP for OSA-Express Gigabit and Fast Ethernet QDIO adapters.

The /usr/lpp/tcpip/samples directory contains the agent capabilities statement, which indicates which MIBs are supported by the SNMP agent and subagents shipped with CS for z/OS 1.2, including any variations on our support, such as read/write objects supported only as read-only. The /usr/lpp/tcpip/samples directory also contains the enterprise-specific MIB definitions.

You will see the list of all the MIB objects supported by CS for z/OS V1R2 in the appendix of *z/OS V1R2 Communications Server: IP User's Guide and Commands*, SC31-8780.

For more information on the SNMP framework, refer to *Managing OS/390 TCP/IP with SNMP*, SG24-5866, or *TCP/IP Tutorial and Technical Overview,* GG24-3376, and the corresponding RFCs.

Part 2

z/OS management tools

3

Network management with SNMP

This chapter describes the detailed configuration information of the SNMP components shipped with CS for z/OS 1.2. The following SNMP components will be discussed in this chapter:

- SNMP agent
- UNIX SNMP manager
- TCP/IP SNMP subagent
- OMPROUTE SNMP subagent
- SLA SNMP subagent
- Web Server subagent

This chapter contains the following sections:

3.1 SNMP overview

The Simple Network Management Protocol (SNMP) protocol allows for architected management type actions and information.

The network management framework for TCP/IP-based internetworks consists of the following entities:

- SNMP managers

  An SNMP manager is an application at a network management terminal that typically issues request operations (the GET and SET requests) to an agent using the SNMP protocol. For CS for OS/390 V2R10 IP this can be the **osnmp** command, the NETVIEW SNMP command, and any other manager in a TCP/IP network.

  SNMP commands such as the **osnmp** command and the NETVIEW SNMP command can be used to access MIB object information. They send SNMP requests to SNMP agents on either local or remote hosts. The **osnmp** command supports SNMPv1, SNMPv2c, and SNMPv3. The NETVIEW SNMP command supports only SNMPv1.

  Note: The SNMP command that can be issued in the UNIX System Services shell environment is provided. It has exactly the same function and syntax as the **osnmp** command.

- SNMP agents

  An SNMP agent is a server at a managed host that responds to the SNMP protocol operations. An agent must be present at each IP host in order to enable management of that host by an SNMP manager. An SNMP agent is basically a server that provides support for the Management Information Base objects that are pertinent to the IP host at which it resides.

- SNMP subagents

  Commonly, an SNMP agent does not maintain the values of all MIB objects directly. These values are generally maintained or accessed through one or more subagents. An SNMP subagent interacts with an agent to respond to the GET and SET requests for a particular set of objects it owns.

  The CS for z/OS 1.2 SNMP agent supports the Distributed Protocol Interface (DPI), an interface by which subagents can communicate with the CS for z/OS 1.2 SNMP agent to support particular MIB objects.

  The following SNMP subagents are shipped with CS for z/OS 1.2:

  - The TCP/IP subagent reports information about the TCP/IP stack.
  - The OMPROUTE subagent reports the information specific to OSPF.
  - The SLA Subagent reports information about defined service policies and performance statistics related to traffic using those policies.
  - Various other products use subagents, for example, IBM’s Web server and OSA/SF.

  At ITSO Raleigh, we used the SLA Subagent to monitor the performance statistics controlled by the CS for z/OS 1.2 Service Policy agent.

See Figure 3-1 on page 21 for an overview diagram of the CS for z/OS 1.2 SNMP support.

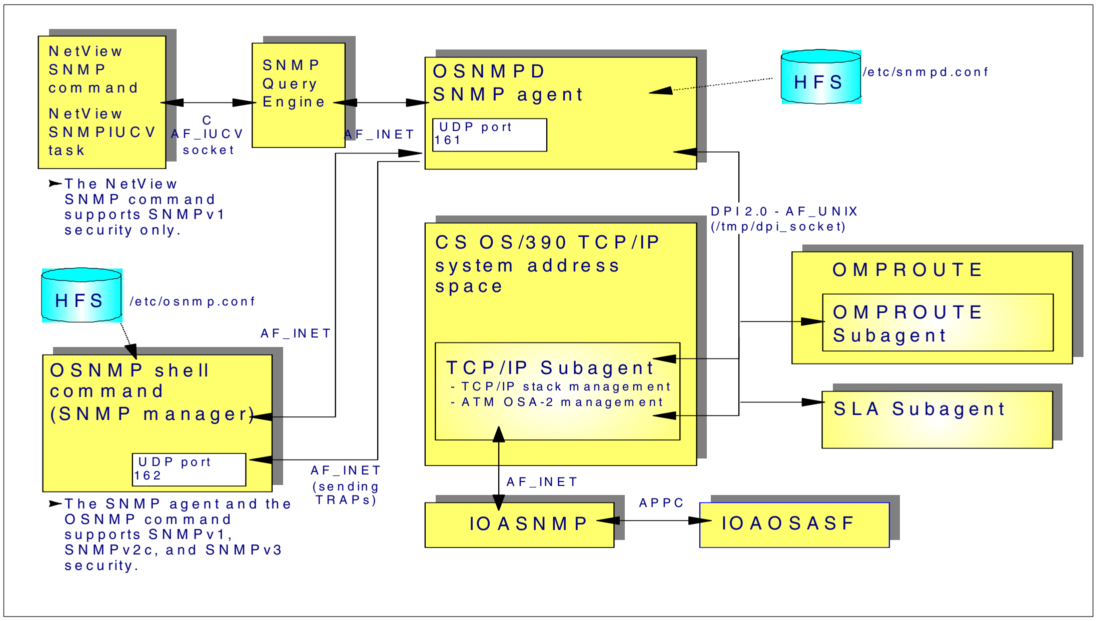


Figure 3-1 SNMP support in CS for z/OS 1.2

The SNMP manager (the **osnmp** command) and SNMP agent shipped as part of CS for z/OS 1.2 support all of the three versions of the SNMP protocol. Note, however, that the NETVIEW SNMP command supports SNMPv1 only.

The overview of SNMP versions is below:

- SNMPv1

  SNMPv1 is defined in RFC 1155 and RFC 1157 and is the initial version of SNMP that is supported by TCP/IP for MVS V3R2. SNMPv1 provides community-based security and supports the following five protocol data units (PDUs):

  - GET
  - GETNEXT
  - SET
  - RESPONSE
  - TRAP

- SNMPv2

  SNMPv2 adds additional SNMP operations such as GETBULK, provides more granular error responses, and defines an additional ability to support an enhanced MIB definition, called SMIv2.

  The SNMPv2 protocol and MIB structure are defined in RFC 1902 through RFC 1908, which in August 1993 became a proposed standard with the status of elective. There are two choices of security defined by RFCs with SNMPv2. One is named SNMPv2c, which supports community-based security like SNMPv1, and the other is SNMPv2u, which provides user-based security and allows installations to define user security names, authentication, and privacy keys. SNMPv2u was defined as an experimental protocol and became a precursor to SNMPv3.

  CS for z/OS 1.2 supports SNMPv2c but not SNMPv2u.

> **Note:** CS for OS/390 supported SNMPv2u in V2R5 and V2R6. That support was dropped in V2R7 because the standards-based SNMPv3 support was introduced.

- SNMPv3

  SNMP Version 3 (SNMPv3), defined in RFCs 2570 through 2575 issued in April 1999, is the standards-based solution to the previous weaknesses of SNMP security. The SNMPv3 architecture is modularized so that portions of it can be enhanced over time without requiring the entire architecture to be replaced. SNMPv3 defines a framework that consists of (among other things):

  - A message processing model (SNMPv3)
  - A security model (user-based security)
  - An access control model (view-based access control)

  The framework is structured so that multiple models can be supported concurrently and replaced over time. For example, although there is a new message format for SNMPv3, messages created with the SNMPv1 and SNMPv2 formats can still be supported. Similarly, the user-based security model can be supported concurrently with community-based security models previously used.

  SNMPv3 provides the ability to configure the agent dynamically from either a local or remote host, and to make changes in the configuration while the SNMP agent is running. Doing SNMP agent configuration dynamically requires a good understanding of how the SNMP SET commands can be issued to create new rows or to change or delete existing rows, and a familiarity with the SNMP engine configuration tables defined in RFCs 2570 through 2575. (For additional details, see RFCs 2570 through 2575, as well as RFCs 1901 through 1910.)

Note: In April 1999 RFCs 2271 through 2275 were obsoleted by RFCs 2571 through 2575.

The Management Information Base defines the objects that may be managed for each layer in the TCP/IP protocol. There are two versions: MIB-I and MIB-II. MIB-I was defined in RFC 1156 and is now classified as a historic protocol with a status of not recommended.

Each managed node supports only those groups that are appropriate. For example, if there is no gateway, the EGP group need not be supported. But if a group is appropriate, all objects in that group must be supported. The list of managed objects defined has been derived from those elements considered essential. This approach of taking only the essential objects is not restrictive, since the SMI provides extensibility mechanisms such as definition of a new version of the MIB and definition of private or nonstandard objects.

In Table 3-1 you will see the summary of RFCs that define the MIB objects supported by CS for z/OS 1.2. Please refer to the RFCs listed for the complete definition.

Table 3-1 SNMP MIB support RFC summary

MIB group	RFC number
Interfaces	RFC 2233
IP	RFC 2011
IP Forwarding Table	RFC 1354 (obsoleted by RFC 2096)
ICMP	RFC 2011
TCP	RFC 2012
UDP	RFC 2013
OSPF	RFC 1850
Ethernet-like Interface	RFC 2665
IP over ATM	RFC 2320
ATM Interface	RFC 1695 (obsoleted by RFC 2515)
IBM OSA Express Interface	IBM TCP/IP MVS Enterprise Specific MIB
DVIPA	IBM TCP/IP MVS Enterprise Specific MIB
SNMP	RFC 1907, RFC 2271-RFC 2275
DPI	RFC 1592
Service Level Agreement Management	RFC 2758 (experimental RFC)
IBM 3172	IBM 3172 Enterprise Specific MIB
IBM Remote Ping	IBM TCP/IP MVS Enterprise Specific MIB
IBM TCP/IP MVS	IBM TCP/IP MVS Enterprise Specific MIB

Support for the RFC 2665 Ethernet-like interface MIB was added in CS for OS/390 V2R10 IP for OSA-Express Gigabit and Fast Ethernet QDIO adapters and vastly improved in z/OS 1.2.

Support for the Dynamic VIPA interface MIB was added in CS for z/OS 1.2.

The /usr/lpp/tcpip/samples directory contains the agent capabilities statement, which indicates which MIBs are supported by the SNMP agent and subagents shipped with CS for z/OS 1.2, including any variations on our support, such as read/write objects supported only as read-only. The /usr/lpp/tcpip/samples directory also contains the enterprise-specific MIB definitions.

You will see the list of all the MIB objects supported by CS for z/OS 1.2 in the appendix of *z/OS V1R2 Communications Server: IP User's Guide and Commands*, SC31-8780.

For more information on the SNMP framework, refer to the following sources:

- *Managing OS/390 TCP/IP with SNMP*, SG24-5866
- *TCP/IP Tutorial and Technical Overview*, GG24-3376
- The corresponding RFCs

3.2 Enhancements introduced by CS for z/OS 1.2

In this section we describe the SNMP enhancements introduced in CS for z/OS 1.2.

The following enhancements were introduced with CS for z/OS 1.2.

- SNMP Subagents RACF support.
  - If you create a resource called EZB.SNMPAGENT.sysname.tcp_procname in the RACLISTed SERVAUTH class, all subagents on that system creating subagents in that stack will require READ access to that resource.
  - If you do not activate this resource, as before, any UID 0 task can create a subagent, possibly overwriting MIB definitions from another subagent.
- More performance counters. The IP stack in z/OS supports performance counters for TCP and IP protocols that are not defined in the standard MIBs. Support has been added to SNMP for these counters in the IBM MVS TCP/IP enterprise-specific MIB, and SIOCGMONDATA ioctl() has also been updated to return this additional information.
  - /usr/lpp/tcpip/samples/mvstcpip.mib contains the formal SNMPv1 syntax of the Enterprise Specific MIB.
  - /usr/lpp/tcpip/samples/mvstcpip.mi2 contains the formal SNMPv2 syntax of the Enterprise Specific MIB.
  - The interface packet counters for the interface MIB (RFC 2232) have been updated to 64 bit.
  - Many more TCP connection-based counters are now in the MIB. Examples are the server backlog and accept counters.
- SNMP_COMMUNITY statement in snmp.conf.

  The new SNMP_COMMUNITY statement maps between v1 and v2 community strings and v3 message parameters. For the time being, the COMMUNITY statement will still be supported and will work as before. It is recommended that you change to using the new format.
- Dynamic and remote update support for the SNMP configuration.
  - The SNMP Community MIB has been added. It contains the following:
    - snmpCommunityIndex
    - snmpCommunityName
    - snmpCommunitySecurityName
    - snmpCommunityContextEngineID
    - snmpCommunityContextName
    - snmpCommunityTransportTag
    - snmpCommunityStorageType
    - snmpCommunityStatus
  - The SNMP target address extension table MIB has been added. It contains the following:
    - snmpTargetAddrTMask
    - snmpTargetAddrMMS
- TARGET_ADDRESS statement has been enhanced to support a set of addresses.
- Enhanced OSA-Express MIB.
  - The Gigabit Fast Ethernet and ATM155 adapters MIB are now in the IBM MVS TCP/IP Enterprise MIB and contain much more information.

  - As before, to get the information you must start OSA/SF and IOASNMP, the devices must be online, and the SACONFIG statement in the IP profile must have the keywords OSAENABLED and OSASF port number specified.
  - The new tables provided are:
    - osaexpChannelTable replaces ATM channel data previously in osasfChannelTable
    - osaexpPerfTable
    - osaexpEthPortTable
    - osaexpEthSnaTable
  - Performance data in channel and performance tables is only available on the zSeries 900 processor microcode Level 1.31 and OSA/SF must have OW45237 applied. See *OS/390 V2R10.0 IP IBM Communications Server User's Guide*, GC31-8514, for a complete list of the supported MIB objects.

- Added MIB for Dynamic VIPA in the IBM MVS TCP/IP Enterprise Specific MIB. This support has been realized through the TCP/IP subagent.
  - ibmMvsDVIPATable DVIPA table
  - ibmMvsDVIPARangeConfTable VIPARANGE table
  - ibmMVSDVIPADistConfTable VIPADISTRIBUTED table
  - ibmMVSDVIPAConnRoutingTable DVIPA connection routing table
  - ibmMVSDVIPADistPortTable DVIPA distributed port table
  - VIPASMPARMS
  - DVIPA traps
    - DVIPA Status Change (will be sent if bit 1 in ibmMVSDVIPATrapControl is on)
    - DVIPA Removed (bit 2)
    - Target Added (bit 3)
    - Target Removed (bit 4)
    - Target Server Started (bit 5)
    - Target Server Ended (bit 6)
  - DVIPA trap control

    The bits in ibmMVSDVIPATrapControl configure which traps are generated. The default is 1 and 2.
  - New TCP Listener Table by connection ID (Resource ID) ibmMVSTcpListenerTable
    - Local IP address - Local Port, Remote IP address - Remote Port
    - Accept count, Backlog values, and Server procedure name

- New cold start trap.

  ibmMVSTcpipSubagentColdStart, is generated if the TCP/IP subagent is restarted, for example, for reconfiguration

- New Server Resource ID entry in TCP Listener Table.

  A new entry in the existing ibmMvsTcpConnTable, ibmMVSTcpConnServerResourceID, has been created, representing the numeric identification of the server for Load Balancing Servers using SHAREPORT. This can be used to index into the new TCP Listener Table.

3.3 SNMPv3 enhancements

SNMPv3 support was introduced in OS/390 V2R7 IP, and CS for OS/390 V2R10 IP made major enhancements to that support, as discussed in the following sections.

3.3.1 UTF-8 support

ISO/IEC 10646-1 defines a multi-octet character set called the Universal Character Set (UCS), which includes most of the world's writing systems. Two multi-octet encodings are defined; UCS-4 (a four-octet per character encoding) and UCS-2 (a two-octet per character encoding).

UTF-8 (UCS transformation format, 8 bit) is an ASCII-compatible unicode. It encodes UCS-2 or UCS-4 characters as a varying number of octets, where the number of octets, and value of each, depends on the integer value assigned to the character in ISO/IEC 10646. The US ASCII (7-bit characters with sign bit off) is considered a proper subset of UTF-8.

In SNMPv3 all the configuration parameters required to configure the agent are represented as MIB variables. This helps to configure the agent dynamically using SET requests. Some variables within these MIBs are defined with the type SnmpAdminString that is basically an octet string that supports UTF-8. In CS for OS/390 V2R10 IP and higher, these strings are allowed to be set using any character set. In previous releases, SETs for these objects to strings containing UTF-8 characters were rejected.

Refer to RFC 2279 for details about the UTF-8 transformation.

3.3.2 Inform support at the SNMP agent

The SNMP agent may send unsolicited notification messages to SNMP manager to report occurred events. There are two types of notification messages, trap and inform.

Traps are unsolicited messages send by the SNMP agent indicating that an event had occurred. They carry unconfirmed class PDUs; the SNMP agent does not expect a response back.

Inform messages are unsolicited messages, like traps, but they carry a confirmed class PDU; the SNMP agent expects to see a response for the inform messages that it sends out.

CS for OS/390 V2R10 IP added the support for the generation of the inform messages.

A change to the existing NOTIFY statement in SNMPD.CONF was made to add the support for inform messages.

Note: The **osnmp** UNIX command does not support inform messages; if an inform message is received, **osnmp** just throws it away and does not respond to it.

3.3.3 Notifications filtering

SNMPv3 allows a user to configure the destinations to which notifications are to be forwarded. Adding notification filtering at the agent will allow an SNMP agent to selectively send notifications that an SNMP manager is interested to receive, which will prevent unnecessary messages and possible performance degrades.

The two tables, snmpNotifyFilterTable and snmpNotifyFilterProfileTable, support notification filtering.

Table 3-2 summarizes the MIB object tables that contain information being used during the SNMPv3 operations. All information in these tables can be configured in the SNMP agent configuration file and updated through the SNMP SET operation.

Table 3-2 MIB object tables used for the SNMPv3 operation

SNMPD.CONF entry	MIB object	Description
USM_USER	usmUserTable	The table of users configured to the SNMP agent
VACM_GROUP	vacmsecurityToGroupTable	This table maps a combination of securityModel and securityName into groupName, which is used to define an access control policy for a group of users
VACM_VIEW	vacmViewTreeFamilyTable	Locally held information about families of subtrees within MIB views
VACM_ACCESS	vacmAccessTable	The table of access rights for groups
NOTIFY	snmpNotifyTable	This table is used to select management targets that should receive notifications, as well as the type of notification that should be sent to each selected management target
TARGET_ADDRESS	snmpTargetAddrTable	A table of transport addresses to be used in the generation of SNMP messages
TARGET_PARAMETERS	snmpTargetParamsTable	A table of SNMP target information to be used in the generation of SNMP messages
NOTIFY_FILTER_PROFILE	snmpNotifyFilterProfileTable	Used to associate a notification filter profile with target parameters

See *z/OS V1R2 Communications Server: IP Configuration Reference*, SC31-8776, for SNMP configuration file entry, and *z/OS V1R2 Communications Server: IP User's Guide and Commands*, SC31-8780, for the RFC-related MIB object.

The following is an example of the filter definitions in the SNMPD.CONF file:

```
#-------------------------------------------------------------------------------------
# NOTIFY entries
# Format is:
#  notifyName tag type storageType
#--------------------------------------------------------------------------------------
NOTIFY notify1 traptag trap -
NOTIFY notify2 informtag inform -       1

#--------------------------------------------------------------------------------------
# TARGET_ADDRESS
# Format is:
#  targetAddrName tDomain tAddress tagList targetParams timeout retryCount storageType
#--------------------------------------------------------------------------------------
```

```
TARGET_ADDRESS Target1 UDP 9.24.104.42     traptag trapparms1 - - -
TARGET_ADDRESS Target2 UDP 9.24.104.149:1160 traptag trapparms2 - - -
TARGET_ADDRESS Target3 UDP 127.0.0.1       traptag trapparms3 - - -
TARGET_ADDRESS Target4 UDP 127.0.0.1       informtag informparms - - - 2

#-------------------------------------------------------------------------------------
# TARGET_PARAMETERS
# Format is:
#  paramsName mpModel securityModel securityName securityLevel storageType
#-------------------------------------------------------------------------------------
TARGET_PARAMETERS trapparms1 SNMPv1   SNMPv1  publicv1  noAuthNoPriv -
TARGET_PARAMETERS trapparms2 SNMPv2c  SNMPv2c publicv2c noAuthNoPriv -
TARGET_PARAMETERS trapparms3 SNMPv3   USM     u28        AuthNoPriv -
TARGET_PARAMETERS informparms SNMPv3  USM     u39        noAuthNoPriv - 3

#-------------------------------------------------------------------------------------
# NOTIFY_FILTER_PROFILE
# Format is:
#  targetParamsName profileName storageType
#-------------------------------------------------------------------------------------
NOTIFY_FILTER_PROFILE trapparms3  filTrap    -
NOTIFY_FILTER_PROFILE informparms filInform  -                     4

#-------------------------------------------------------------------------------------
# NOTIFY_FILTER
# Format is:
#  profileName filterSubtree filterMask filterType storageType
#-------------------------------------------------------------------------------------
NOTIFY_FILTER filTrap    authenticationFailure - included -
NOTIFY_FILTER filInform  Interfaces inform included -              5
```

We defined notify1 and notify2 to send notification messages. Notify2 1 will send inform messages to a Network Manager defined in the Target4 2 entry, and use the target parameter defined in informparms 3 to configure securityModel, securityName, and securityLevel. We used the filter profile filInform 5, defined via informparms 4 in NOTIFY_FILTER_PARAMETER, to define which inform messages should be notified. We decided to send all inform messages related to the MIB object of interfaces.

3.3.4 Environment variables for configuration files

You can use the environment variable, MIBS_DATA, for use with the **osnmp** command, and the environment variables PW_SRC and SNMPTRAP_DEST for use with the SNMP agent. With this support, the location of any SNMP agent configuration file can be configured via an environment variable. Table 3-3 summarizes all environment variables used by SNMP entities, which are the SNMP agent, manager, and trap forwarder daemon.

Table 3-3 Environment variables supported by SNMP entities

Environment variable	Configuration file for	Default location in HFS
SNMP agent		
SNMPD_BOOTS	SNMP agent boot file	/etc/snmpd.boots
SNMPD_CONF	SNMP agent configuration	/etc/snmpd.conf
OSNMPD_DATA	SNMP agent MIBs	/etc/osnmpd.data
SNMPTRAP_DEST	SNMPTRAP.DEST statement	/etc/snmptrap.dest
PW_SRC	PW.SRC statement	/etc/pw.src

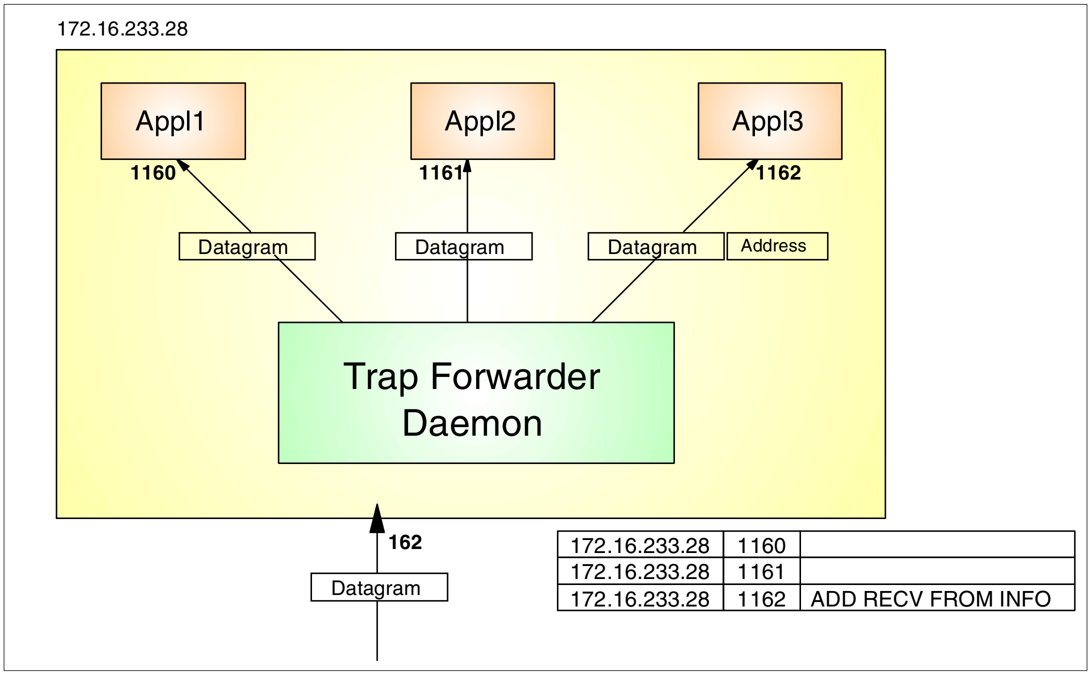
Environment variable	Configuration file for	Default location in HFS
SNMP manager		
OSNMP_CONF	`osnmp` configuration	/etc/osnmp.conf
MIBS_DATA	MIBS.DATA information	/etc/mibs.data
Trap forwarder daemon		
TRAPFWD_CONF	TRAPFWD daemon configuration	/etc/trapfwd.conf

3.3.5 Multiple SNMP manager's support on well-known port

SNMPv3 allows you to configure the port numbers over which notification messages are sent so that multiple SNMP managers over the same TCP/IP stack may run. However, most SNMPv1 agents do not support the customized port numbers, but use the hard-coded port number, which is the well-known UDP port 162, to send trap messages.

You cannot start more than one SNMP manager at the same time to listen on a well-known port for the same IP address; to alleviate the port contention problem so that multiple SNMP manager applications can receive the traps sent on well-known port 162, the trap forwarder is required.

The trap forwarder listens on well-known port 162 for trap messages and forwards the trap messages to all the SNMP managers defined in the TRAPFWD.CONF file.



172.16.233.28	1160	
172.16.233.28	1161	
172.16.233.28	1162	ADD RECV FROM INFO

Figure 3-2 Trap forwarder

In Figure 3-2 you see the trap forwarder defined to listen on port 162 and forward the trap messages received to other SNMP manager applications Appl1, Appl2, and Appl3 on a configured port.

The trap forwarder can be used to forward the trap messages to SNMP managers on the same host or other hosts.

Trap forwarder implementation

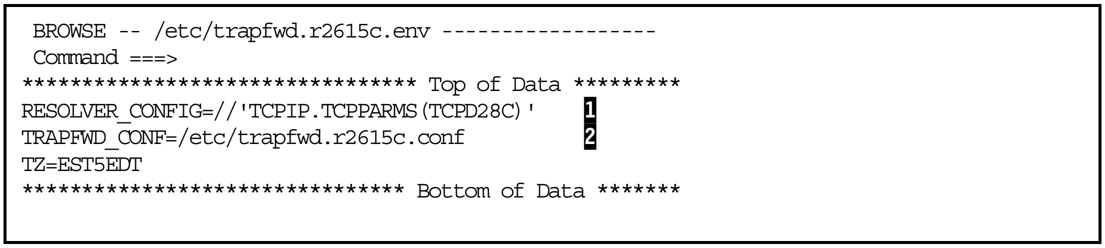
While trap forwarder can be started as an MVS procedure or as a UNIX shell program, we configured it as an MVS started procedure. The following is the started procedure we used:

```
//TRAPFWD   PROC
//*
//* Sample procedure for running the OE Trap Forwarder daemon
//* IBM Communications Server for OS/390
//*         Licensed Materials - Property of IBM
//*         Status = CSV2R10
//*
//TRAPFWD EXEC PGM=EZASNTRA,REGION=4096K,TIME=NOLIMIT,
//   PARM='POSIX(ON) ALL31(ON) ENVAR("_CEE_ENVFILE=DD:STDENV")/-d 0' 1
//*
//STDENV   DD PATH='/etc/trapfwd.r2615c.env',PATHOPTS=(ORDONLY)      2
//*
//SYSPRINT DD SYSOUT=*,DCB=(RECFM=F,LRECL=80,BLKSIZE=80)
//SYSIN    DD DUMMY
//SYSERR   DD SYSOUT=*
//SYSOUT   DD SYSOUT=*,DCB=(RECFM=F,LRECL=80,BLKSIZE=80)
//CEEDUMP  DD SYSOUT=*
```

We started the TRAPFWD with no trace activated 1, but you can use several options:

- -d n, which indicates a debugging level, and the valid values are:
  - 0: No tracing.
  - 1: Minimal tracing. Traces address from which the trap is received.
  - 2: In addition to 1. Traces address to which the packet is forwarded.
  - 3: in addition to 2. Traces trap packets.
- -p port_number, which defines the UDP port at which the trap forwarder should listen for traps. The default is UDP port 162.
- -l max_packet_len, which indicates the maximum packet length of the trap datagram that has forwarded. The valid values are 4096 (4 KB) to 16384 (16 KB). The default value is 4096.

We used a STDENV DD statement 2 to point to an HFS file that contains our environment variables for trap forwarder. Figure 3-3 shows our environment variable settings in /etc/trapfwd.r2615c.env.

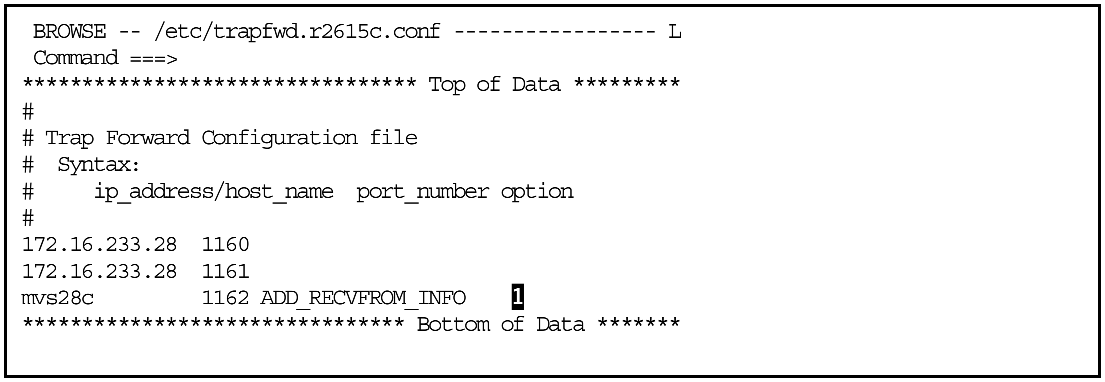


```
 BROWSE -- /etc/trapfwd.r2615c.env ------------------
 Command ===>
********************************* Top of Data *********
RESOLVER_CONFIG=//'TCPIP.TCPPARMS(TCPD28C)'     1
TRAPFWD_CONF=/etc/trapfwd.r2615c.conf           2
TZ=EST5EDT
******************************** Bottom of Data *******
```

Figure 3-3 Environment variable definitions for the trap forwarder

We used RESOLVER_CONFIG 1 to associate the TRAPFWD with a specific TCP/IP stack, and the TRAPFWD_CONF 2 environment variable to refer a trap forwarder configuration file.

In the trap forwarder configuration file we defined the IP address and port number for where to forward the trap messages. See Figure 3-4 for our sample configuration defined in /etc/trapfwd.r2615c.conf.



```
 BROWSE -- /etc/trapfwd.r2615c.conf ----------------- L
 Command ===>
********************************* Top of Data *********
#
# Trap Forward Configuration file
#  Syntax:
#     ip_address/host_name  port_number option
#
172.16.233.28  1160
172.16.233.28  1161
mvs28c         1162 ADD_RECVFROM_INFO    1
******************************** Bottom of Data *******
```

Figure 3-4 Trap forwarder configuration file

If you define the option 1 ADD_RECVFROM_INFO, the IP address from which the trap is received is appended to the trap. The address information contains the IP address of the trap message originator followed by the length of the structure. The default is not to append the address information. You would only want to use ADD_RECVFROM_INFO to send to a manager that can parse the information from the end of the packet (neither **osnmp** nor the SNMP query engine do).

To interface with the trap forwarder daemon, the following MODIFY commands were introduced:

- **MODIFY trapfwd,TRACE,QUERY**, which displays the level of tracing set for the trap forwarder daemon.
- **MODIFY trapfwd,TRACE,LEVEL=n**, which sets the level of tracing to the level specified.
- **MODIFY trapfwd,REFRESH**, which refreshes the configuration dynamically.

See *z/OS V1R2 Communications Server: IP Configuration Guide*, SC31-8775, and *z/OS V1R2 Communications Server: IP Configuration Reference*, SC31-8776, for more details on trap forwarder implementation.

3.3.6 Source IP of UDP packets sent by the agent and manager

The **osnmp** command and the z/OS SNMP Agent use SOURCEVIPA by default. If you want to use the physical interface address then use the -a start option on the agent or the NOSVIPA command in the **onmp** command. If you are using unrouted IP addresses for your interfaces, do not use this option. In a walk command, z/OS's **osnmp** command will use the IP address returned in the data to build the next request, regardless of what you specified on the -h parameter.

Note that the TCP/IP subagent uses the IP address of the primary interface, which will be retrieved using the gethostid API, to connect to the SNMP agent. The primary interface is the IP address defined in the PRIMARYINTERFACE statement or the first entry in the HOME list if PRIMARYINTERFACE is not specified.

3.4 Configuring the SNMP agent (OSNMPD) and subagent

Prior to configuring the SNMP agent, you have to decide on your security needs, that is, the community-based security using SNMPv1/SNMPv2c or the user-based security provided by the SNMPv3 framework. At ITSO Raleigh we implemented the SNMPv3 security level without privacy.

To configure the SNMP agent and subagents, follow the steps below:

1. Configure TCPIP.PROFILE.
2. Manually configure the SNMPD.BOOTS file, if needed.
3. Specify MIB objects and their values in the OSNMPD.DATA file.
4. Create authentication keys, if you are using the SNMPv3 security level.
5. Configure the SNMP agent and the security environment using the SNMPD.CONF file.
6. Start the SNMP agent and verify the log information.
7. Verify the configuration using the **osnmp** command. You need to have the /etc/osnmp.conf file customized.

> **Note:** You can use the SNMP UNIX command instead of the **osnmp** command.

For more information about the SNMP agent and subagents' configuration, please refer to *Managing OS/390 TCP/IP with SNMP*, SG24-5866.

3.4.1 TCPIP.PROFILE configuration

For the SNMP agent we reserved port 161, and for the **osnmp** command port 162 and 16200 to receive the trap information. The port reservation is configured in TCPIP.PROFILE.

```
; Member TCPIP.TCPPARMS(PROF39A)
;**********************************************************************
PORT
   161 UDP OSNMPD              ; SNMP Agent
   162 UDP OMVS                ; osnmp command (SNMP Manager)
 16200 UDP OMVS                ; osnmp command (SNMP Manager)
```

If the agent is started from the OS/390 shell, reserve port 161 for OS/390 UNIX by typing `OMVS` instead of the MVS procedure name. If you use NETVIEW as a manager you should reserve port 162 to the SNMP query engine or leave it unreserved.

For the TCP/IP subagent configuration, two statements are available in TCPIP.PROFILE, which are ITRACE and SACONFIG. ITRACE is used to turn on or off the internal trace for the TCP/IP subagent. This function should be used only when requested by an IBM service representative.

The SACONFIG is the main statement for the SNMP agent configuration. If SACONFIG is *not* defined, the following default values are set:

- COMMUNITY public.

  The TCP/IP subagent communicates with the CS for OS/390 SNMP agent by the SNMPv1 protocol with the community name configured in the COMMUNITY keyword. Therefore, the community name specified on this keyword must match one that is defined in the PW.SRC or SNMPD.CONF data set used by the SNMP agent or specified on the -c parameter when the SNMP agent is started. The default value is public.
- Default port 161 to initiate communication with the SNMP agent.

- The TCP/IP subagent is started at TCP/IP initialization.
- The OSA ATM Management was not used.

We have the following statements configured:

```
; -------------------------------
SACONFIG  COMMUNITY MVSsubagent 1
 ENABLED
 AGENT 161
 SETSENABLED    2
; OSASF 760
; OSAENABLED
; SETSDISABLED
```

For the OMPROUTE subagent, we have the ROUTESA_CONFIG statement configured in the OMPROUTE configuration data set:

```
ROUTESA_CONFIG ENABLED=YES
             COMMUNITY="MVSsubagent"; 3
```

1 , 3 The community name MVSsubagent was chosen to establish communication between the SNMP agent and each SNMP subagent in CS for OS/390 IP.

We enabled the SET command at the TCP/IP subagent 2. You will see the complete lists of MIBs that CS for z/OS 1.2 supports in the Appendix in *z/OS V1R2 Communications Server: IP User's Guide and Commands,* SC31-8780.

3.4.2 SNMPD.BOOTS

The SNMP agent uses the SNMPD.BOOTS configuration file to support SNMPv3 security. This file contains agent information used to authenticate the SNMPv3 requests. The SNMPD.BOOTS keeps the engine ID of the SNMP agent and the number of times the agent reboots. If no SNMPD.BOOTS file exists when the agent is started, the agent creates one. If a file does exist, the agent uses the values specified in the file for setting its engine ID and engineBoots values. If the file exists but contains invalid values for engine ID or engineBoots, the agent issues a message and terminates. This file contains two parameters:

- Engine ID, which uniquely identifies, in an administrative domain, the SNMP agent.
- engineBoots, which is a count of the number of times the SNMP agent has rebooted/reinitialized since the last configuration was made.

The engine ID contained in SNMPD.BOOTS is interpreted as follows:

```
00000002          00000000xxxxxxxx
```

The first four octets are set to the private enterprise number for IBM, and the value is 2. The remaining eight octets are specified in an enterprise-specific way. The SNMP agent in CS for OS/390 V2R10 IP uses its IP address, which is given by gethostname and gethostbyname API, for this field.

From CS for OS/390 V2R10 IP onwards the **osnmp** command uses the VIPA address as its own address. **osnmp** does gethostname to retrieve the host name of the TCP/IP stack on which it is running, and then issues the gethostbyname API, which uses the Domain Name Server or the host file configured locally to retrieve the IP address associated with that host name. Therefore, the VIPA used is not necessarily configured as the primary IP address.

You can just as easily use an arbitrary engine ID, such as 1234. As long as you used that engine ID when creating localized keys for use at that SNMP agent, that engine ID will work just as well as the one the SNMP agent would have created. You, however, should take care that there are no duplicate engine IDs within the administrative domain.

Localized keys used for SNMPv3 security should be derived using the engine ID.

The search order of SNMPD.BOOTS information is as follows:

- The name of an HFS file or an MVS file specified by the SNMPD_BOOTS environment variable
- /etc/snmpd.boots

We implemented the SNMP agent and subagent on system RA28 and RA39 at ITSO Raleigh. We configured /etc/osnmpd.r2615a.boots in both systems to use the VIPA IP addresses as shown in Table 3-4.

Table 3-4 SNMPD.BOOTS definitions in our test environment

System	VIPA IP address	Boots file
RA28A	172.16.252.28	0000000200000000AC10FC1C 0000000001
RA39A	172.16.232.39	0000000200000000AC10E827 0000000001

3.4.3 OSNMPD.DATA

OSNMPD.DATA contains values for selected MIB objects. You can customize those values for your environment. If no OSNMPD.DATA file is found, the defaults for these MIB objects are as shown in Table 3-5.

Table 3-5 OSNMPD.DATA: MIBs' default values

Object name	Default value
sysDescr	The value of the HOSTNAME environment variable, if configured, or the OS/390 system identifier under which the agent is running.
sysContact	SNMPBASE-Unspecified.
sysLocation	SNMPBASE-Unspecified.
sysName	SNMPBASE-Unspecified.
sysObjectId	1.3.6.1.4.1.2.3.13.
sysServices	A single octet that defaults to 0.
snmpEnableAuthenTraps	Default value is 2, which means authentication traps are disabled.
saDefaultTimeout	Five seconds.

The first file found in the following search order will be used to obtain the OSNMPD.DATA information.

1. The name of an HFS file or MVS file specified by the OSNMPD_DATA environment variable
2. /etc/osnmpd.data HFS file

3. The data set specified on the OSNMPD DD statement in the agent procedure
4. *jobname*.OSNMPD.DATA, where *jobname* is the name of the job used to start the SNMP agent
5. SYS1.TCPPARMS(OSNMPD)
6. hlq.OSNMPD.DATA, where hlq either defaults to TCP/IP or is specified on the DATASETPREFIX statement in the TCPIP.DATA file being used

OSNMPD picks up the HOSTNAME from TCPDATA and translates this name using a domain name server into its agent address. We used the following OSNMPD.DATA definitions:

```
#
# osnmpd.data sample
#
# Sample file for setting MIB variables and options for
# the SNMPv3 Agent provided by OS/390
#
# Licensed Materials - Property of IBM
# "Restricted Materials of IBM"
# 5647-A01
# (C) Copyright IBM Corp. 1996, 2000
# Status = CSV2R10
#
  sysDescr "SNMPv3 agent version 1.0 with DPI version 2.0"
  sysContact  "SNMP Administrator"
  sysLocation "IBM ITSO Raleigh NC"
  sysName "CS for z/OS V1R2 System PORK"
# Default value of sysObjectID is equivalent to ibmTcpIpMvs
#  in the ibmAgents subtree; this is the sysObjectID representing
#  IBM Communications Server for OS/390
  sysObjectID     "1.3.6.1.4.1.2.3.13"
  snmpEnableAuthenTraps 1
  saDefaultTimeout    6
  saMaxTimeout  700
# saAllowDuplicateIDs must be set to 1 to allow multiple DPI version 1
#  subagents
  saAllowDuplicateIDs   1
  dpiPathNameForUnixStream "/tmp/dpi_socket"
# Default value of sysServices indicates support for
#  internet, end-to-end, and application layers as
#  defined in RFC 1907.
  sysServices      76
```

The default value of sysObjectID is equivalent to ibmTcpIpMvs in the ibmAgents subtree; this is the sysObjectID representing IBM Communications Server for OS/390 IP Services.

3.4.4 Authentication process and key generation

Authentication is required to process SNMPv3 requests (unless the requested security level is noAuth). The authentication protocols available are HMAC-MD5 and HMAC-SHA, described in RFC 2574.

The authentication process consists of:

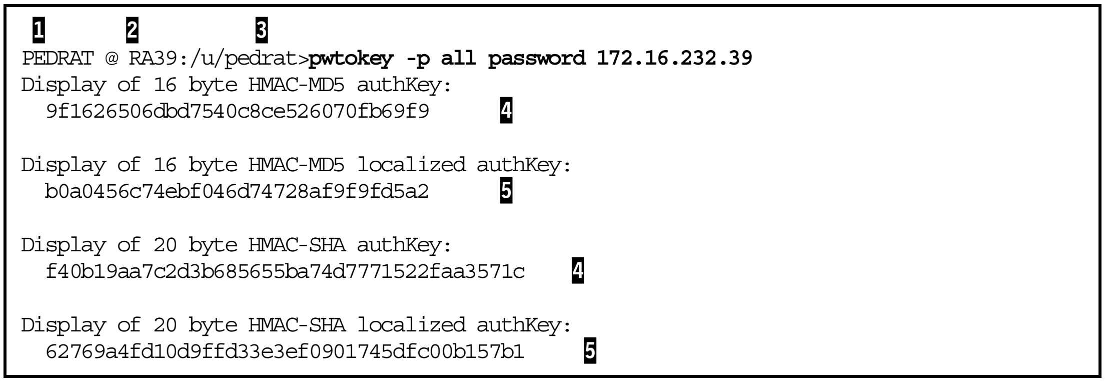
- Verification of the integrity of a received message
- Verification of the user on whose behalf the message was generated

Note: An SNMP engine must have knowledge of that user, so a match is done between the userName and authKey specified in the osnmp.conf and snmpd.conf files.

The facility pwtokey converts passwords into authentication keys. It takes as input the password and an identifier of the agent (the engine ID or the IP address).

The output generates two kinds of keys: One that is localized (usable only in the specified agent) and one that is not localized. It is recommended that you use localized keys in an SNMP agent configuration and nonlocalized keys in an SNMP manager configuration, that is, in /etc/snmpd.conf and /etc/osnmp.conf, respectively, in most installations.

Figure 3-5 is a sample of the pwtokey output obtained in our implementation.



```
 1      2         3
PEDRAT @ RA39:/u/pedrat>pwtokey -p all password 172.16.232.39
Display of 16 byte HMAC-MD5 authKey:
  9f1626506dbd7540c8ce526070fb69f9      4

Display of 16 byte HMAC-MD5 localized authKey:
  b0a0456c74ebf046d74728af9f9fd5a2      5

Display of 20 byte HMAC-SHA authKey:
  f40b19aa7c2d3b685655ba74d7771522faa3571c    4

Display of 20 byte HMAC-SHA localized authKey:
  62769a4fd10d9ffd33e3ef0901745dfc00b157b1     5
```

Figure 3-5 pwtokey utility

1 indicates we want to generate keys with both HMAC-MD5 and HMAC-SHA protocols.

2 is the password we use in our implementation.

3 is the agent IP address that was used to create the engine ID. If you choose to use your own algorithm for creating an engine ID, you have to invoke the **pwtokey** command with the -e flag and the engine ID instead of the IP address.

4 is the nonlocalized key used in the osnmp.conf file for the SNMP manager.

5 is the localized key used in the snmpd.conf file for the SNMP agent.

Note: In our implementation we did not use privacy keys because they are used for encryption purposes only. Encryption support for SNMPv3 requests is now provided as part of the base FMID. In previous releases, a separate feature FMID was required.

3.4.5 Configuring the SNMP agent

The SNMPD.CONF file defines the SNMP agent security and notification destinations. If the SNMPD.CONF file exists, the agent can support SNMPv1, SNMPv2c, and SNMPv3 requests. If no SNMPD.CONF file exists, the agent will support only SNMPv1 and SNMPv2c requests.

To access SNMPD.CONF information, the search order is:

1. An HFS file or an MVS file specified by the SNMPD_CONF environment variable
2. /etc/snmpd.conf

We configured the SNMP agent to support SNMPv3, SNMPv2c, and SNMPv1 requests, so we put our definitions in the /etc/snmpd.conf file.

The SNMPv3 allows you to configure the agent dynamically, from either a local or remote host, and to make changes while the agent is running. The dynamic configuration changes are written out in the snmpd.conf file so it remains in effect in case the agent is restarted.

If you want to use only community-based security (SNMPv1 and SNMPv2c) without the view-based access model, you may put your definitions in the PW.SRC and SNMPTRAP.DEST files. You can still use the SNMPD.CONF file for just SNMPv1 and SNMPv2c security, but it is more complicated to set up. Therefore, you will probably prefer to use the legacy PW.SRC and SNMPTRAP.DEST files.

> **Note:** If the SNMPD.CONF file is found, the PW.SRC and SNMPTRAP.DEST files will not be used.

The 11 types of entries described in Table 3-6 could be configured in the SNMPD.CONF file.

Table 3-6 The SNMPD.CONF entries

Entry name	Description
USM_USER	Defines a user for the User-Based Security Model (USM).
VACM_GROUP	Defines a security group (made up of users or communities) for the View-Based Access Control Model (VACM).
VACM_VIEW	Defines a particular set of MIB objects, called a view, for the View-Based Access Control Model.
VACM_ACCESS	Identifies the access permitted to different security groups for the View-Based Access Control Model.
NOTIFY	Identifies management targets to receive notifications.
NOTIFY_FILTER_PROFILE	Used to associate a notification filter profile with target parameters.
NOTIFY_FILTER	Contains all the filter profiles that are used to determine whether a management target should receive a particular notification.
TARGET_ADDRESS	Defines a management application's address and identifies parameters to be used in sending notifications.
TARGET_PARAMETERS	Defines the message processing and identifies security parameters to be used in sending notifications to a particular management target.
COMMUNITY	Defines a community for community-based security.
DEFAULT_SECURITY	Identifies the default security posture to be configured for the SNMP agent.

The following configuration file was used to configure the SNMP agent on system RA28A in our environment:

```
# osnmpd.r2615a.conf--   configuration file for the SNMP agent
#
```

```
# Licensed Materials - Property of IBM
# Status = CSV2R10
#
#---------------------------------------------------------------------------
# USM_USER entries    1
# Format is:                                10
#  userName engineID authProto authKey privProto privKey keyType storageType
#---------------------------------------------------------------------------
USM_USER  u1 - HMAC-MD5 d82f936bd31fef9dac3d5bb0803c63c0  - - L -
USM_USER  u2 -  none       -                              - - L -
USM_USER  u03 - HMAC-MD5 b0a0456c74ebf046d74728af9f9fd5a2 - - L -
USM_USER  u28 - HMAC-MD5 b0a0456c74ebf046d74728af9f9fd5a2 - - L -
USM_USER  u39 - HMAC-MD5 b0a0456c74ebf046d74728af9f9fd5a2 - - L -
                                  11                          12
#--------------------------------------------------------------------------
# VACM_GROUP entries    2
# Format is:                              10
#  groupName securityModel securityName storageType
#-------------------------------------------------------------------------
VACM_GROUP group1 USM     u1 -
VACM_GROUP group1 USM     u2 -

VACM_GROUP group1 USM     u03 -
VACM_GROUP group1 USM     u28 -  11
VACM_GROUP group1 USM     u39 -

VACM_GROUP group3 SNMPv1  publicv1  -
VACM_GROUP group3 SNMPv2c publicv2c -    12

VACM_GROUP group4 SNMPv1  MVSsubagent  -
VACM_GROUP group4 SNMPv2c MVSsubagent  -    13

#------------------------------------------------------------------------
# VACM_VIEW entries    3
# Format is:                                10
#  viewName viewSubtree viewMask viewType storageType
#-----------------------------------------------------------------------
VACM_VIEW bigView        internet   - included -

VACM_VIEW prettyBigView  internet   - included -
VACM_VIEW prettyBigView  interfaces - excluded -

VACM_VIEW mediumView     system     - included -
VACM_VIEW mediumView     interfaces - included -
VACM_VIEW mediumView     tcp        - included -
VACM_VIEW mediumView     udp        - included -
VACM_VIEW mediumView     icmp       - included -

VACM_VIEW smallView      snmp       - included -

VACM_VIEW subagentView   dpiPort    - included -  16

#------------------------------------------------------------------------------------------
-
# VACM_ACCESS entries    4
# Format is:                                                                           10
#           group context context security security  read     write        notify  storage
#            Name  Prefix  Match   Level    Model     View     View         View     Type
#------------------------------------------------------------------------------------------
-
```

```
VACM_ACCESS  group1 - - AuthPriv     USM     bigView        bigView      bigView 14 -
VACM_ACCESS  group1 - - AuthNoPriv   USM     bigView        prettyBigView bigView 14 -
VACM_ACCESS  group1 - - noAuthNoPriv USM     smallView      smallView     smallView -

VACM_ACCESS  group3 - - noAuthNoPriv SNMPv1  mediumView     mediumView    mediumView -
VACM_ACCESS  group3 - - noAuthNoPriv SNMPv2c bigView        bigView       bigView  15 -

VACM_ACCESS  group4 - - noAuthNoPriv SNMPv1  subagentView   -             -         - 16
VACM_ACCESS  group4 - - noAuthNoPriv SNMPv2c subagentView   -             -         - 16

#------------------------------------------------------------------------------------------
# NOTIFY entries   5
# Format is:             10
#  notifyName tag type storageType
#-----------------------------------------------------------------------------------------
NOTIFY notify1 traptag trap -
*NOTIFY notify2 informtag inform -

#-----------------------------------------------------------------------------------------
# TARGET_ADDRESS   6
# Format is:                                                                    10
#           target  tDomain tAddress          tagList  target  timeout retry  storage
#           AddrName                                 Params         Count   Type
#-----------------------------------------------------------------------------------------
                   19          17
#TARGET_ADDRESS Target1 UDP 9.67.113.10     traptag trapparms1 - - -
TARGET_ADDRESS Target2 UDP 172.16.250.3:16200 traptag trapparms2 - - -
                               18
TARGET_ADDRESS Target3 UDP 127.0.0.1       traptag trapparms3 - - -
*TARGET_ADDRESS Target4 UDP 127.0.0.1       informtag informparms - - -

#------------------------------------------------------------------------------------------
# TARGET_PARAMETERS   7
# Format is:                                     10
#                params    mpModel security security   security   storage
#                 Name             Model    Name       Level      Type
#------------------------------------------------------------------------------------------
TARGET_PARAMETERS trapparms1 SNMPv1   SNMPv1  publicv1  noAuthNoPriv -
TARGET_PARAMETERS trapparms2 SNMPv2c  SNMPv2c publicv2c noAuthNoPriv -
TARGET_PARAMETERS trapparms3 SNMPv3   USM     u2        AuthNoPriv -
*TARGET_PARAMETERS informparms SNMPv3  USM     u2        noAuthNoPriv -

#------------------------------------------------------------------------------------------
# NOTIFY_FILTER_PROFILE
#------------------------------------------------------------------------------------------
NOTIFY_FILTER_PROFILE trapparms3  filProf    -

#------------------------------------------------------------------------------------------
# NOTIFY_FILTER
#------------------------------------------------------------------------------------------
NOTIFY_FILTER filProf  authenticationFailure - included -

#------------------------------------------------------------------------------------------
# COMMUNITY     8
# Format is:                                                               10
#        community    security    security       netAddr    netMask  storageType
#         Name         Name         Level
#------------------------------------------------------------------------------------------
COMMUNITY publicv1    publicv1     noAuthNoPriv 0.0.0.0     0.0.0.0         -
COMMUNITY publicv2c   publicv2c    noAuthNoPriv 0.0.0.0     0.0.0.0         -
COMMUNITY MVSsubagent MVSsubagent  noAuthNoPriv 0.0.0.0     0.0.0.0        -

#---------------------------------------------------------------------------------------------
# DEFAULT_SECURITY    9
# Format is:
#  securityPosture password privacy
#---------------------------------------------------------------------------------------------
DEFAULT_SECURITY semi-secure defaultpassword no

#---------------------------------------------------------------------------------------------
# Any SNMP agent configuration entries added by dynamic configuration
# (SET) requests get added to the end of the SNMPD.CONF file.
#---------------------------------------------------------------------------------------------
```

Section 1 describes the users for the user security model (USM). The name must be unique in the agent configuration and it is associated to a key that is used in the authentication process following the protocol specified. A localized key created by the pwtokey utility is recommended to be used for this entry. When you use a nonlocalized key, you have to specify `N` for the keyType keyword.

Section 2 associates the user name previously defined (or generically, the SNMPv1 and SNMPv2c community) to a group name. We use three groups; one is for the SNMPv3 security domain and all members belong to group1 11. For SNMPv1 and SNMPv2 community domain, group3 is configured 12, and each SNMP subagent will be a member of group4 13. The community name for MSV subagents, MVSsubagent in this case, has to match the value in the subagent configurations.

Section 3 defines the view name associated with the MIB object prefix of the MIB tree in the view. Valid values are both textual object IDs (OIDs) (for example, Internet) or numerical (for example, 1.3.6.1.2.1...).

Section 4 associates the group name to the views previously defined, stating the security level and the authentication protocol to be used. For example, a user belonging to group1 with a AuthNoPriv security level can read all MIB Internet information but cannot set (write) the interface values.

For the SNMP subagents that are members of group4, we allow them to read the view subagentView, which has been defined in the VACM_VIEW statement 16.

Section 5 defines the notification information. CS for z/OS 1.2 supports both trap and inform notification; see 3.3.3, “Notifications filtering” on page 26.

Section 6 defines the notification target(s). It should be an SNMP manager (for example, Tivoli NETVIEW), which will receive the generated traps (usually on port 162) to take the appropriate actions. In our environment, the SNMP agent will send traps to the **`osnmp`** command running on the system RA03A (its IP address is 172.16.250.3) 17 over UDP port number 16200, and the local host (127.0.0.1) 18.

Section 7 defines the security and the message processing model to be used in sending the notifications to the previously defined targets.

The index of the TARGET_ADDRESS statement is the targetAddrName 19 for the traps. If you need to send traps to one more different address, then the targetAddrName for each destination should be different.

For traps to be sent, the user should be granted access to the part of the subtree that is required to send out traps. In our sample configuration, the TARGET_PARAMETERS statement shown below specifies that the trap should be sent on behalf of user u2.

```
TARGET_PARAMETERS trapparms3 SNMPv3   USM u2  AuthNoPriv -
```

Because user u2 belongs to group1, in order to receive traps, group1 has to have access to bigView for notifyView **14**. Note that smallView can access only the SNMP group and the traps are not forwarded. Therefore, the VACM_ACCESS statement should be configured as follows:

```
VACM_ACCESS  group1 - - AuthNoPriv  USM  bigView  prettyBigView bigView -
```

This will grant the access required to send the traps.

Similarly, for traps in the SNMPv1 and SNMPv2 domains, the groups have to have access to bigView to notify View in our configuration **15**.

Section **8** describes the community security environment for requests coming from SNMP managers that do not support (or do not want to use) the SNMPv3 security model.

Section **9** defines the security default values to be applied to the previous entries. DEFAULT_SECURITY is intended to help customers to get started without adding all the other definitions. It defines some default users, views, and access rights.

The parameter **10** indicates the type of storage in which the definitions are to be maintained. The default taken is nonvolatile, which means the information will persist across reboots of the agent, but it can be changed by dynamic configuration requests.

For more information about syntax and definition, please refer to:

- *z/OS V1R2 Communications Server: IP User's Guide and Commands*, SC31-8780
- *z/OS V1R2 Communications Server: IP Configuration Guide*, SC31-8775
- *z/OS V1R2 Communications Server: IP Configuration Reference*, SC31-8776

3.4.6 Working with SNMP agent and subagents

Before you bring up and test the SNMP agent, please make sure the syslog daemon (syslogd) is started.

Syslogd is a server process that has to be started as one of the first processes in your UNIX System Services environment. The SNMP agent uses syslogd for logging purposes, and to send trace information to it. If syslogd is not started, log and trace data appears on the MVS console.

Starting SNMP agent

The SNMP agent can be started as either a UNIX System Services shell command or MVS started task. Starting the agent from MVS may be required for a system automation purpose. Either way, the SNMP agent has to have super user authority.

We started the SNMP agent from the MVS environment using the procedure below:

```
//OSNMPD    PROC
//*
//* Communications Server for OS/390
//*         Licensed Materials - Property of IBM
//*         Status = CSV2R10
//*
//OSNMPD EXEC PGM=EZASNMPD,REGION=4096K,TIME=NOLIMIT,
//   PARM=('POSIX(ON) ALL31(ON)',
//   'ENVAR("_CEE_ENVFILE=DD:STDENV")/-d 0')
//*
//*
//STDENV   DD PATH='/etc/osnmpd.r2615a.env',PATHOPTS=(ORDONLY)  1
//SYSPRINT DD SYSOUT=*,DCB=(RECFM=F,LRECL=80,BLKSIZE=80)
//SYSIN    DD DUMMY
//SYSERR   DD SYSOUT=*
//SYSOUT   DD SYSOUT=*,DCB=(RECFM=F,LRECL=80,BLKSIZE=80)
//CEEDUMP  DD SYSOUT=*
```

OSNMPD must find the name (TCPIPJOBNAME in TCPIP.DATA) with which it should be associated. You can set the environment variable RESOLVER_CONFIG to point to the correct resolver file when multiple INET physical file systems are started. If only one INET PFS is started then we encourage you to use the /etc/resolv.conf file. The SYSTCPD DD card is also supported by OSNMPD.

We use an HFS file 1 to define the **osnmp** environment variables, as shown in Figure 3-6.

```
RESOLVER_CONFIG=//'TCPIP.TCPPARMS(TCPD39A)'
SNMPD_CONF=/etc/osnmpd.r2615a.conf                  1
SNMPD_BOOTS=/etc/osnmpd.r2615a.boots                2
OSNMPD_DATA=/etc/osnmpd.r2615a.data                 3
TZ=EST5EDT
```

Figure 3-6   Environment variable definitions for the SNMP agent - /etc/osnmpd.r2615a.env

We have following three environment variables defined to locate files related to the SNMP agent’s configuration:

- SNMPD_CONF 1 for the SNMP agent configuration file. See Figure 3-5 on page 36 for our sample configuration.
- SNMPD_BOOTS 2 for the SNMP boots file. The values we have configured are shown in Table 3-4 on page 34.
- OSNMPD_DATA 3 for the SNMP agent’s MIB definitions. Our definition can be found in Figure 3-2 on page 29.

Figure 3-7 shows the MVS START command and the system responses.

```
S OSNMPD
$HASP100 OSNMPD   ON STCINRDR
IEF695I START OSNMPD   WITH JOBNAME OSNMPD   IS ASSIGNED TO USER TCPIP3
 , GROUP OMVSGRP
$HASP373 OSNMPD   STARTED
IEF403I OSNMPD - STARTED - TIME=11.40.58
EZZ6225I SNMP AGENT: INITIALIZATION COMPLETE
EZZ3218I SNMP SUBAGENT: CONNECTED TO OSA/SF            1
EZZ3202I SNMP SUBAGENT: INITIALIZATION COMPLETE        2
EZZ3221I SNMP SUBAGENT: SET REQUESTS ENABLED
EZZ8101I OMPROUTE SUBAGENT INITIALIZATION COMPLETE     3
```

Figure 3-7   SNMP agent start messages

1 The TCP/IP subagent has connected to OSA/SF, and the communication to the SNMP agent has been established 2. The OMPROUTE subagent has also established a communication path to the SNMP agent 3.

When OSNMPD is started with debug level 15, the following log messages will tell you the MVS subagent has connected to the SNMP agent successfully. Those messages will be sent to the syslog daemon and written in its log file.

```
SNMP logging data follows =============================
 Log_type:          snmpLOGrequest_in
    origin:         UDP 127.0.0.1 port 1049
 version:           SNMPv1        1
 community:         MVSsubagent 2
  ..in hex:         4d 56 53 73 75 62 61 67 65 6e 74
 addressInfo:       UDP 127.0.0.1 port 1049    3
 PDUtype:           GetRequest ('a0'h)
 request:           1
 error-status:      noError (0)
 error-index:       0
 varBind oid:
 OBJECT_IDENTIFIER
 1.3.6.1.4.1.2.2.1.1.3.0
 1.3.6.1.4.1.2.2.1.1.3.0
         name:      dpiPathNameForUnixStream.0
         value:
 NULL

 End of SNMP logging data: =============================
 SNMP logging data follows =============================
 Log_type:          snmpLOGresponse_out
    send rc:        0
    destination:    UDP 127.0.0.1 port 1049
 version:           SNMPv1
 community:         MVSsubagent
  ..in hex:         4d 56 53 73 75 62 61 67 65 6e 74
 addressInfo:       UDP 127.0.0.1 port 1049
 PDUtype:           GetResponse ('a2'h)
 request:           1
 error-status:      noError (0)
 error-index:       0
 error-index:       0
 varBind oid:
 OBJECT_IDENTIFIER
 1.3.6.1.4.1.2.2.1.1.3.0
         name:      dpiPathNameForUnixStream.0
         value:
 OCTET_STRING
 (DisplayString)
 /tmp/dpi_socket               4
 End of SNMP logging data: =============================
 EZZ6246I Accepted new DPI UNIX socket connection on fd=7
 pDPIpacket: Major=2, Version=2, Release=0, Id=0, Type=SNMP_DPI_OPEN
 pDPIopen: subagent Identification=1.3.6.1.4.1.2.11.7.2    5
           Description=OS/390 TCP/IP SNMP Subagent         5
           Selected Character Set is 0 (Native)
           timeout=120, max_varBinds=0, password=** NONE **
```

Figure 3-8 SNMP agent log message (OS/390 TCP/IP SNMP subagent)

1 By sending the SNMPv1 GET request PDU, the SNMP subagent obtains the UNIX stream socket name 4 to communicate with the SNMP agent using AF_UNIX connections. For this request, the community name of MVSsubagent 2, which has been configured for the SACONFIG statement in TCPIP.PROFILE, is used. The IP address that the MVS subagent is using is also shown in this log message 3. This IP address has to match the COMMUNITY

definition in the SNMPD.CONF file for the SNMP subagent to connect to the agent. When the subagent opens an AF_UNIX connection with the SNMP agent, the subagent's identification is written 5. Once the stream socket name is received back from the agent as an SNMPv1 response, the community name will not be used for anything else.

In the case of the OMPROUTE subagent and SLA Subagent, the identification will be different, as shown in Figure 3-9 and Figure 3-10.

```
EZZ6246I Accepted new DPI UNIX socket connection on fd=8
 pDPIpacket: Major=2, Version=2, Release=0, Id=0, Type=SNMP_DPI_OPEN
 pDPIopen: subagent Identification=1.3.6.1.4.1.2.11.7.3
           Description=OS/390 TCP/IP OSPF SNMP Subagent
           Selected Character Set is 0 (Native)
           timeout=120, max_varBinds=0, password=** NONE **
```

Figure 3-9 SNMP agent log message (OS/390 OSPF SNMP subagent)

```
DPIpacket: Major=2, Version=2, Release=0, Id=0, Type=SNMP_DPI_OPEN
DPIopen: subagent Identification=1.3.6.1.4.1.2.11.7.4
         Description=Pagtsnmp SLAPM-MIB DPI Subagent
         Selected Character Set is 0 (Native)
         timeout=120, max_varBinds=0, password=** NONE **
```

Figure 3-10 SNMP agent log message (OS/390 SLA Subagent)

3.4.7 Configure z/OS SNMP manager

CS for z/OS 1.2 provides two different SNMP manager functions, that is the **osnmp** command and the Tivoli NETVIEW SNMP command. We decided to use the **osnmp** command in our test, because only the **osnmp** command supports the SNMPv3 operations.

Note: Starting with CS for OS/390 V2R10 IP, the SNMP command is supported in the UNIX shell environment. The command syntax is identical to that of the **osnmp** command.

SNMP manager configuration

To issue SNMPv3 requests, the **osnmp** command needs the OSNMP.CONF file which is used to define target agents and, for SNMPv3, the security parameters to be used in sending requests to them.

The following search order for OSNMP.CONF enables different copies of the file to be used by different users:

1. An HFS file or MVS data set pointed to by the OSNMP_CONF environment variable
2. /etc/osnmp.conf
3. /etc/snmpv2.conf

In our installation, we have the OSNMP_CONF environment variable set as follows in the user's $HOME/.profile:

```
export OSNMP_CONF=/etc/osnmpM.r2615a.conf
```

The /etc/osnmpM.r2615a.conf HFS file is used by the **osnmp** command.

```
#---------------------------------------------------------------------------------------
--
#
# Format of entries:
#   winSnmp  target  admin  sec  password  context sec   auth  authKey  privProto
privKey
#     Name    Agent         Name                   Level Proto
#      1        2     3      4       5              6      7      8
#---------------------------------------------------------------------------------------
----
#---------------------------------------------------------------------------------------
----
# Community-based security  (SNMPv1 and SNMPv2c)
#---------------------------------------------------------------------------------------
----
v1       127.0.0.1   snmpv1
v2c      127.0.0.1   snmpv2c
v2cm39a  9.24.104.149 snmpv2c
#
#---------------------------------------------------------------------------------------
-----
# User-based Security Model (USM with SNMPV3)

#---------------------------------------------------------------------------------------
------
#   For SNMP agent on system RA28
ura28a  172.16.252.28 snmpv3 u1 - - AuthNoPriv   HMAC-MD5
561928c07349b7d68e1850888948934d - -
ura28   172.16.252.28 snmpv3 u2 - - noAuthNoPriv - - - - -
mvs28a  172.16.252.28 snmpv3 u28 - - AuthNoPriv  HMAC-MD5
9f1626506dbd7540c8ce526070fb69f9 - -
#
#   For SNMP agent on system RA39
ura39a  172.16.232.39 snmpv3 u1 - - AuthNoPriv   HMAC-MD5
3ff41bdb5dfe48e3cf08f6be130c84d9 - -
ura39   172.16.232.39 snmpv3 u2 - - noAuthNoPriv - - - - -
mvs39a  172.16.232.39 snmpv3 u28 - - AuthNoPriv  HMAC-MD5
9f1626506dbd7540c8ce526070fb69f9 - -
#
```

Figure 3-11 SNMP manager configuration file on RA28

The definitions are:

1 The label by which **osnmp** identifies an entry in this table. This name is specified in the -h command option.

2 The target agent IP address or name.

3 Specifies the administrative model adopted by the agent. Valid values are snmpv1, snmpv2c, and snmpv3.

4 The user name. This name must be known by the agent.

5 The password used to generate the key (see 3.4.4, “Authentication process and key generation” on page 35). If a password is specified, it is used to automatically generate any needed keys and the authKey and privKey fields below are ignored.

Note: The use of passwords instead of keys in this configuration file is *not* recommended, as storing passwords in this file is less secure than using keys.

6 The security level requested.

7 The SNMP authentication protocol used (see 3.4.4, “Authentication process and key generation” on page 35).

8 The authentication key generated for this user.

In *z/OS V1R2 Communications Server: IP Configuration Reference*, SC31-8776, you will find a detailed description of all parameters available for the **osnmp** configuration.

If you want to use textual names for any MIB objects that are not defined in any compiled MIB, you can define them in an /etc/mibs.data HFS file.

Using the osnmp command

The following sample screens should give you an idea of what you can do with the **osnmp** command.

You can find the **osnmp** command syntax and parameter description in *z/OS V1R2 Communications Server: IP User’s Guide and Commands*, SC31-8780.

The Management Information Base objects are listed in the appendix in *z/OS V1R2 Communications Server: IP User’s Guide and Commands*, SC31-8780.

The table shows various objects. The objects and information included are the object descriptor; object identifier; if the object is supported by the agent or subagent, where the object is defined (RFC number, subagent MIB, etc.); and the access allowed (read-only or read/write).

You can find the non-standard MIB definitions in the HFS directory /usr/lpp/tcpip/samples.

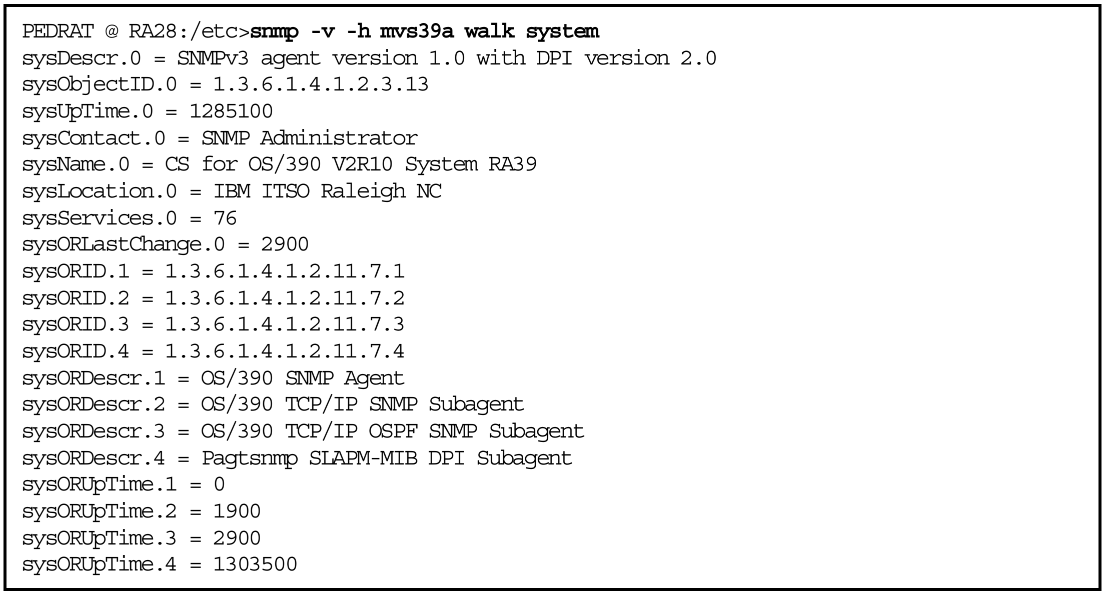
You might need to debug your SNMP **osnmp** command environment. To do this, you must activate debug levels 1 to 4. The log information is normally displayed on your screen. To capture the log information in a file, you can redirect output into an HFS file as follows:

```
osnmp -d 1 -h v3m39a walk system > /tmp/osnmp.log
```

All messages will be recorded in the HFS file specified in the redirection command (>). If the file you specified does not exist, the system creates this file automatically.

Throughout our test we used SNMPv3 authentication, so the -h parameter identified the label name defined in /etc/osnmpd.r2615a.conf (see the example in “Starting SNMP agent” on page 41) and the community name was not specified because it was not used.

Figure 3-12 on page 47 shows the system display for our SNMP agent on system RA39. We provided some of this information in the agent configuration HFS file /etc/osnmpd.r2615a.data (see Figure 3-12 on page 47).



```
PEDRAT @ RA28:/etc>snmp -v -h mvs39a walk system
sysDescr.0 = SNMPv3 agent version 1.0 with DPI version 2.0
sysObjectID.0 = 1.3.6.1.4.1.2.3.13
sysUpTime.0 = 1285100
sysContact.0 = SNMP Administrator
sysName.0 = CS for OS/390 V2R10 System RA39
sysLocation.0 = IBM ITSO Raleigh NC
sysServices.0 = 76
sysORLastChange.0 = 2900
sysORID.1 = 1.3.6.1.4.1.2.11.7.1
sysORID.2 = 1.3.6.1.4.1.2.11.7.2
sysORID.3 = 1.3.6.1.4.1.2.11.7.3
sysORID.4 = 1.3.6.1.4.1.2.11.7.4
sysORDescr.1 = OS/390 SNMP Agent
sysORDescr.2 = OS/390 TCP/IP SNMP Subagent
sysORDescr.3 = OS/390 TCP/IP OSPF SNMP Subagent
sysORDescr.4 = Pagtsnmp SLAPM-MIB DPI Subagent
sysORUpTime.1 = 0
sysORUpTime.2 = 1900
sysORUpTime.3 = 2900
sysORUpTime.4 = 1303500
```

Figure 3-12 Display of the system MIB with -v option

By using the -v option for the **osnmp** command, you can display the output from a request in textual name instead of the MIB object identifier. This option might help you to understand the MIB information more easily.

3.4.8 Configure SLA Subagent

The Service Level Agreement (SLA) subagent shipped in CS for z/OS 1.2 provides the following services:

- Retrieves performance monitoring data from the stack and allows monitoring of various QoS characteristics
- Provides traps for monitored entities (policies/applications) that violate boundaries established using the PolicyMonitorTable MIB

We run the SLA Subagent on the system RA28 and RA39 using the started procedure shown below. The sample start procedure is shipped as TCPIP.SEZAINST(PAGTSNMP).

```
//PAGTSNMP PROC
//*
//* SecureWay Communications Server IP
//* 5647-A01 (C) Copyright IBM Corp. 1999.
//* Licensed Materials - Property of IBM
//*
//PAGTSNMP EXEC PGM=PAGTSNMP,REGION=OK,TIME=NOLIMIT,         1
//       PARM='POSIX(ON) ALL31(ON) ENVAR("_CEE_ENVFILE=DD:STDENV")/-c M
//             VSsubagent -t 30'
//*            VSsubagent -d 2 -t 30'
//*
//STDENV   DD PATH='/etc/pagtsnmp.r2615a.env',PATHOPTS=(ORDONLY) 3
//*STDENV   DD DSN=TCPIP.TCPPARMS.R2615(PAGT&SYSCLONE.A),DISP=SHR
//*
//SYSPRINT DD SYSOUT=*  2
//SYSOUT   DD SYSOUT=*  2
//*
//CEEDUMP  DD SYSOUT=*,DCB=(RECFM=FB,LRECL=132,BLKSIZE=132)
```

SLA Subagent uses the SNMPv1 protocol to communicate with the SNMP agent to obtain the UNIX stream socket name 1, through which the communication between the agent and manager will be established. The same community name that has been defined in the SNMP agent configuration file has to be specified for the SLA Subagent.

Output written to stdout and stderr goes to the data set or file specified with SYSPRINT or SYSOUT, respectively. 2 But normally, PAGTSNMP does not write output to stdout or stderr. Instead, output is written to syslogd. When the -o parameter is specified, however, output is written to stdout instead of syslogd.

In the HFS file specified by the STDENV DD 3, we configured the environment variables as shown below:

```
BROWSE -- /etc/pagtsnmp.r2615a.env ----------------
 Command ===>
********************************* Top of Data ******
RESOLVER_CONFIG=//'TCPIP.TCPPARMS(TCPD39A)'
LIBPATH=/usr/lib:     1
******************************** Bottom of Data ****
```

We had to define the LIBPATH 1 for find PAPI.DLL. If you do not have this DLL available, the PAGTSNMP will write the following error messages in SYSOUT and fail:

```
IEF472I PAGTSNMP PAGTSNMP - COMPLETION CODE - SYSTEM=806 USER=0000 REASON=000000
CEE3501S The module papi.dll was not found.
From entry point papiConnect at compile unit offset +00000074 at addres
CEE3DMP V2 R10.0: Condition processing resulted in the unhandled condition.
```

When starting the PAGTSNMP, you will see the following message in the MVS console:

```
S PAGTSNMP
$HASP100 PAGTSNMP ON STCINRDR
IEF695I START PAGTSNMP WITH JOBNAME PAGTSNMP IS ASSIGNED TO USER
TCPIP3  , GROUP OMVSGRP
$HASP373 PAGTSNMP STARTED
IEF403I PAGTSNMP - STARTED - TIME=12.45.28
+EZZ8229I PAGTSNMP SUBAGENT: CONNECTED TO PAGENT
+EZZ8216I PAGTSNMP SUBAGENT: CONNECTED TO SNMP AGENT
+EZZ8203I PAGTSNMP SUBAGENT: INITIALIZATION COMPLETE
```

The syslog daemon also writes down some messages from the SLA Subagent. If you set the debug flag, you will see detailed information in the syslogd’s log file.

```
PASubA[16777280]: EZZ8222I Pagtsnmp subagent: running as jobname PAGTSNMP
PASubA[16777280]: EZZ8229I PAGTSNMP SUBAGENT: CONNECTED TO PAGENT
snmpagent[16777260]: EZZ6246I Accepted new DPI UNIX socket connection on fd=9
PASubA[16777280]: EZZ8216I PAGTSNMP SUBAGENT: CONNECTED TO SNMP AGENT
PASubA[16777280]: EZZ8203I PAGTSNMP SUBAGENT: INITIALIZATION COMPLETE
```

To refer to the SNMP agent’s log message, you will also be able to make sure the SLA agent connects to the SNMP agent successfully (see Figure 3-10 on page 44).

```
F PAGTSNMP,QUERY                     2
+EZZ8224I PAGTSNMP SUBAGENT: TRACING IS SET TO 0
+EZZ8219I PAGTSNMP SUBAGENT: MODIFY REQUEST COMPLETED

F PAGTSNMP,TRACE,LEVEL=1             1
+EZZ8219I PAGTSNMP SUBAGENT: MODIFY REQUEST COMPLETED

F PAGTSNMP,QUERY                     2
+EZZ8224I PAGTSNMP SUBAGENT: TRACING IS SET TO 1
+EZZ8219I PAGTSNMP SUBAGENT: MODIFY REQUEST COMPLETED
```

You can change the debug level of the subagent with the MODIFY MVS console command 1. To query the current debug level in effect for the SLA Subagent, you can also use the MODIFY command 2.

3.5 Quick start

As you can see, setting up CS for z/OS 1.2 with the good security and many configuration options that SNMP v3 offers can be a time consuming task. For those of you who are working in a small trusted intranet environment, a single stack, and an existing SNMP v1 infrastructure, the following quick start offers you a chance to get going with SNMP with minimum of configuration changes.

3.5.1 Started task

Take the supplied sample of the SNMP Agent *hlq*.SEZAINST(EZASNDPR) and customize it for your environment. Place the customized version in one of your system proclibs. Use the -c parameter to specify the snmp v1 community string. Do not use “public”. See Example 3-1.

Example 3-1 SNMP Agent started task

```
//TCPSNMA PROC MODULE='EZASNMPD',PARMS='-c myv1pass'
//SNMPD    EXEC PGM=&MODULE,REGION=4096K,TIME=NOLIMIT,
//      PARM='POSIX(ON) ALL31(ON)/&PARMS'
//SYSERR   DD PATH='/tmp/snma.syserr',
//         PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
//         PATHMODE=SIRWXU
//STDOUT   DD PATH='/tmp/snma.stdout',
//         PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
//         PATHMODE=SIRWXU
//STDERR   DD PATH='/tmp/snma.stderr',
//         PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
//         PATHMODE=SIRWXU
//SYSOUT   DD PATH='/tmp/snma.sysout',
//         PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
//         PATHMODE=SIRWXU
//CEEDUMP  DD SYSOUT=*
//*
```

Make an RACF definition to allow the task to run either in the STARTED class or use ICHRIN03 and start the task.

3.5.2 Profile changes

If you want the IP task to start and monitor the SNMP task add the following entries to your IP profile.

- AUTOLOG
- SNMAPROC
- PORT 161 UDP SNMAPROC

Otherwise, just add the information to your automatic operations system.

Add the line `SAFONFIG SETSENABLE COMMUNITY myv1pas` to your IP profile, and create an extra file with a disable command for the SACONFIG, like this:

```
SACONFIG disabled
```

Now activate the SACONFIG disable via vary tcpip,,obey,*filename*, wait for the message confirming the de-activation, and then refresh your profile.

In older releases of OS/390 the community name and the port number were reset to a default value when an OBEYFILE command was used to process the SACONFIG statement. This is no longer the case.

You can now use the SNMP or **osnmp** commands in USS or any other v1 compatible manager in your network to get and set MIB variables in z/OS 1.2 SNMP.

```
PORK:/u/nichols: snmp -v -c myv1pass -h 127.0.0.1 get sysDescr.0
sysDescr.0 = Sysname: OS/390 Nodename: PORK Release: 12.00 Version: 03 Machine:9672
```

3.5.3 Basic NETVIEW integration

If you want to use the panel based NETVIEW SNMP manager to interface to SNMP information, add the data sets to the NETVIEW DD's as indicated in Table 3-7.

Table 3-7 NETVIEW DD cards

PDS/member name	NETVIEW DD
TCPIP.SEZANCLS	DSICLD
TCPIP.SEZANPNL	CNMPNL1
TCPIP.SEZANCLS	DSIPARM
TCPIP.SEZADSIP	DSIPARM

Now customize the supplied SNMP Query agent from *hlq*.SEZAINST(SNMPPPROC) and place it in one of your system proclibs, add the required RACF profile, and start the task.

If you want the IP task to start and monitor the SNMP task add the following entries to your IP profile.

- AUTOLOG
- SNMPPROC
- PORT 162 UDP SNMPPROC

Otherwise just add the information to your automatic operations system.

Add the following line to your task start up table DSIDMN:

- TASK MOD=SNMPIUCV,TSKID=SNMPIUCV,PRI=5,INIT=Y

Edit the *hlq*.SEZADSIP(SNMPARMS) member, changing the SNMPQE name to the name of your stc, in our example, SNMPPROC, and restart NETVIEW to bring in your changes.

You can now use the NETVIEW command SNMPMGMT to get SNMP information from z/OS using a NETVIEW panel interface, or directly from the command line. Example 3-2 shows how it looks when you start the panel interface.

Example 3-2 NETVIEW panel interface

```
SNMPMGM1
X=================================================================X
|                                              Page  1  of  1     |
|                                                                 |
|  TTTTTTTT     CCCCC     PPPPP      XXX    IIIIIIII    PPPPP     |
|     TT       CC    CC   PP   PP   XXX        II       PP   PP   |
|     TT      CC          PP  PP   XXX         II       PP  PP    |
|     TT       CC           PPPPP    XXX          II        PPPPP      |
|     TT        CC   CC     PP      XXX           II        PP         |
|     TT          CCCC      PP     XXX         IIIIIII      PP         |
|                                                                      |
|         SSSS        NNN    NN    MMM     MMM    PPPPP               |
|      SSS    SSS     NNNN   NN    MM MM MM MM    PP   PP             |
|        SSS          NN NNN NN    MM  MMM  MM    PP  PP              |
|          SSS        NN   NNNM    MM   M   MM    PPPPP               |
|      SSS   SSS      NN    NNN    MM       MM    PP                  |
|         SSSS        NN     NN    MM       MM    PP                  |
|                                                                      |
|                                                                      |
|                                                                      |
|                                                                      |
| PF 1=Help      2=Ping      3=Exit      4=Groups   5=Traps   6=Roll  |
| PF 7=Poll Host 8=Poll Var  9=Get/Getn  10=Set                       |
X======================================================================X
```

Press PF4 and you will get some information about the system.

Example 3-3 NETVIEW system information

```
SNMPGRP1
X======================================================================X
|             Query SNMP Group/Table                Page  1  of  2    |
| HOST  127.0.0.1                                                     |
| COMMUNITY NAME                                                      |
| SELECT GROUP  1                                                     |
|                                                                     |
|   1   SYS      - GET SYSTEM GROUP MIB VARIABLES                     |
|   2   IFTAB    - GET INTERFACE TABLE                                |
|   3   ARPTAB   - GET ARP TABLE                                      |
|   4   IP       - GET IP GROUP MIB VARIABLES                         |
|   5   IPADDR   - GET IP ADDRESS TABLE                               |
|   6   IPROUTE  - GET IP ROUTING TABLE                               |
|   7   ICMP     - GET ICMP GROUP MIB VARIABLES                       |
|   8   TCP      - GET TCP GROUP MIB VARIABLES                        |
|   9   TCPTAB   - GET TCP CONNECTION TABLE                           |
|  10   UDP      - GET UDP GROUP MIB VARIABLES                        |
|  11   EGP      - GET EGP GROUP/TABLE                                |
|  12   UDPTAB   - GET UDP LISTENER TABLE                             |
|  13   SNMP     - GET SNMP GROUP                                     |
|  14   IAT      - GET IP ADDRESS TRANSLATION TABLE                   |
|  15   3172SYS  - GET IBM3172 SYSTEM GROUP                           |
| PF 1=Help        3=Main Menu        8=Forward       Enter=Proceed   |
X======================================================================X
```

Note that the community name is not echoed. After entering which group you would like to see, NETVIEW sends the request and receives the response via the SNMP query agent task, displaying it in a nice full screen format, as shown in Example 3-4.

Example 3-4 Displaying a group in NETVIEW

```
SNMPSYS1
 X======================================================================X
 |                                                 Page  1  of  2      |
 |    SYSTEM Group for 127.0.0.1                                       |
 |                                                                     |
 | Descr:    Sysname: OS/390 Nodename: I5HE Release: 12.00 Version: 0  |
 |           3 Machine: 9672                                           |
 | ObjectId: 1.3.6.1.4.1.2.3.13                                       |
 | UpTime:   6 days/22 hours/49 minutes/8 seconds                     |
 | Contact:  SNMPBASE - Unspecified                                   |
 | Name:     SNMPBASE - Unspecified                                   |
 | Location: SNMPBASE - Unspecified                                   |
 | Services: 0                                                        |
 |                                                                    |
 |                                                                    |
 |                                                                    |
 |                                                                    |
 |                                                                    |
 |                                                                    |
 |                                                                    |
 |                                                                    |
 |                                                                    |
 |         PF 8=Forw                                  12 = Return     |
 X====================================================================X
```

As all the panels and REXX EXECs are source supplied, you can expand and customize this manager as you require. You can find full documentation of this feature in the member SNMPREXX of your *hlq*.SEZAINST pds.

If you are wondering how to specify the contact, name, and location, you can customize that in the file /etc/osnmpd.data. There is a example of this file in /usr/lpp/tcpip/samples.

3.5.4 Adding other data sources

In the following sections we discuss adding other data sources.

OSA

You can get native OSA information from your IOANMAIN task by adding the directive OSAENABLED to the SACONFIG directive in your IP profile. The IOASNMP task can be found in *hlq*.SIOASAMP(IOASNMP). Note that the OSAENABLED directive gives you a lot more information than was available before, so its a good thing to add this keyword. See Example 3-5.

Example 3-5 SACONFIG profile statement to retrieve OSA information

```
SACONFIG COMMUNITY aceras OSAENABLED OSASF 2001 SETSENABLED
```

When you activate this you will see the following messages:

```
EZZ3218I SNMP SUBAGENT: CONNECTED TO OSA/SF
EZZ3202I SNMP SUBAGENT: INITIALIZATION COMPLETE
```

You can access many of the new variables under the group ibmTCPIPmvsOsaExpGroup.

IBM Web Server

The Web Server from IBM comes with a ready-to-roll subagent. Just add the directives to your httpd.conf file and restart the Web Server as shown in Example 3-6.

Example 3-6 Turning SNMP on for Web Server

```
#        SNMP directive:
#
#        Set SNMP communication on or off.
#
#        Default:  off
#        Syntax:   SNMP  <on | off>
SNMP   on


#        SNMPCommunity directive:
#
#        The community name which is used by the server to communicate
#        with the SNMP agent.
#
#        Default:  public
#        Syntax:   SNMPCommunity  <public | community name>
SNMPCommunity   myv1pass
```

The following displays lets us find out about the identity of our Web Server.

```
S PTCPWWW
$HASP100 PTCPWWW  ON STCINRDR
IEF695I START PTCPWWW  WITH JOBNAME PTCPWWW  IS ASSIGNED TO USER STCWEB
 , GROUP STCWEBG
$HASP373 PTCPWWW  STARTED
IEF403I PTCPWWW - STARTED - TIME=17.36.31
IMW3534I PID: 50400036 SERVER STARTING
IMW3536I SA 50400036 0.0.0.0:80 * * READY
BPXF024I (STCSNMP) Jul  3 15:36:32 PORK snmpagent 65566 : EZZ6244I 489

Accepted new DPI inet socket connection on fd=7 from 164.32.47.6 port 1230
```

```
osnmp -v -h 127.0.0.1 -c myv1pass get  1.3.6.1.4.1.2.6.120.1.1.1.1.1.1.1

ibmInternetConnSvr.1.1.1.1.1.1.1 = '49424d20485454502053657276657220 2d204e6f72
746820416d65726963616e2045646974696f6e2056563552334d30'h
```

If we can convert that ascii string, it tells us:

```
IBM HTTP Server - North American Edition VV5R3M0-OS/390 PORK
```

Where PORK is our system ID. Much more useful information (for example, about the number of connections, threads, responses given and discarded) is also available. If you want to have reasonable names displayed, update the file /etc/mibs.data with names you like. For example, you could use the entry:

```
myServerid 1.3.6.1.4.1.2.6.120.1.1.1.1.1.1.1. octetstring
```

To get the manager to display:

```
osnmp -v -h 127.0.0.1 -c myv1pass get  myServerid

myServerid = '49424d204854545020536572766572202d204e6f72746820416d65726963616e
2045646974696f6e2056563552334d30'h
```

Computer Associates’ NETSPY product also interfaces to the MIB to get information about IP and combine it with VTAM session data. The configuration is contained in the DD INITPRM. Just add:

```
TCPIPMON=YES
SNMPHOST ADDRESS=127.0.0.1 COMMNAME=myv1pass TIMEOUT=30
```

3.5.5 Other ways of managing your SNMP information

There are a variety of SNMP manager solutions available; IBM's Tivoli provides an enterprise-wide solution, integrating SNA and IP information and displaying it in customizable views. The IP Discovery component (IP Disco) of Tivoli can allow you to quickly produce usable views of your Network, and it has a built-in MIB browser with a GUI. For those of you working in smaller installations or .edu sites with restrictive budgets, there are, of course, freeware alternatives.

You can easily integrate SNMP statistics from CS for z/OS 1.2 into NetSaint, which is a freeware product that runs on most UNIX derivatives including Linux. Several CGI programs are included with this product to allow you to view the results via a Web browser. Reports can be generated, services and service levels monitored, network views generated, and you can be informed of problems, for example, via e-mail.

NetSaint can be downloaded at:

`http://www.netsaint.org`

3.5.6 Sending traps from user processes

Sometimes it can be useful to send a trap from a user process, perhaps to inform an automation product about a state of affairs that is not under an SNMP agent's control. The correct way of doing this is to use the dpi. An example is in /usr/lpp/tcpip/samples/dpi_mvs_sample.c.

For those of you who would like to use REXX, it is possible to construct the UDP packet manually and send the data via the REXX socket interface. This also helps you understand the structure of SNMP packets and the encoding (BER) they use. Example 3-7 shows the format of a v1 trap. It is not an example of how to nicely code such a trap creation program, rather, it is designed to show you the layout of the packet.

Example 3-7 Sample REXX program for sending traps

```
/*REXX*/

data=''
data.=''

data.1='30'x   /* Universal Sequence Start */
data.2='2B'x   /*Length of data - Not Including Me (NIM)*/
data.3='02'x   /*An Integer Follows: Version */
data.4='01'x   /*Length of the data - NIM */
data.5='00'x   /*Version #*/
data.6='04'x   /*Its an Octet String: Community*/
data.7='06'x   /*Length of the data - NIM*/
data.8='70'x   /*p*/
data.9='75'x   /*u*/
data.10='62'x  /*b*/
data.11='6C'x  /*l*/
data.12='69'x  /*i*/
data.13='63'x  /*c*/
data.14='A4'x  /*TRAP-PDU*/
data.15='1E'x  /*Length of (trap) data (including 30 00 terminator)*/
data.16='06'x  /*Object Identifier ASN.1*/
data.17='08'x  /*Length of the data NIM*/
data.18='2B'x  /*1.3 is coded as 1*40+3 = 43 = 0x2b*/
data.19='06'x  /*6 - dod     */
data.20='01'x  /*1 - Internet*/
data.21='04'x  /*4 - private */
data.22='01'x  /*1 - enterprises */
data.23='02'x  /*2 - ibm   */
data.24='03'x  /*3 - ibmagents */
data.25='0D'x  /*13 - ibmtcpipmvs */
data.26='40'x  /*IP Address follows: Client Address */
data.27='04'x  /*Length of the data NIM*/
data.28='A4'x  /*164*/
data.29='19'x  /*25*/
data.30='82'x  /*130*/
data.31='01'x  /*1*/
data.32='02'x  /*Integer: Trap Type*/
data.33='01'x  /*Length NIM */
data.34='04'x  /*4 - Authentication Failed  */
data.35='02'x  /*Integer: Specific Trap Type*/
data.36='01'x  /*Length NIM */
data.37='00'x  /*0 - Nothing*/
data.38='43'x  /*Timeticks: Timestamp*/
data.39='04'x  /*Length NIM */
data.40='12'x  /*Timestamp is 4 bytes binary*/
data.41='34'x  /*Timestamp is 4 bytes binary*/
data.42='56'x  /*Timestamp is 4 bytes binary*/
data.43='78'x  /*Timestamp is 4 bytes binary*/
data.44='30'x  /*Terminated by Universal Sequence 30 00 */
data.45='00'x  /*Terminated by Universal Sequence 30 00 */
data.0 = 45

do i = 1 to data.0
data=data||data.i
end

parse value 'SOCKET'('Initialise','rxtrap') With SRC SubTask Maxdesc Service
parse value 'SOCKET'('Socket',2,'DATAGRAM',0) With SRC SocketID
parse value 'SOCKET'('Sendto',SocketID,Data,,'AF_INET 162 my.host.name') With Src Len
parse value 'SOCKET'('Terminate') with SRC SubTask
exit 0

/*

Primitive ASN.1 Types Identifier in hex
---------------------------------------
INTEGER           02
BIT STRING        03
OCTET STRING      04
NULL              05
OBJECT IDENTIFIER 06

Constructed ASN.1 type Identifier in hex
----------------------------------------
SEQUENCE          30

Primitive SNMP application types Identifier in hex
--------------------------------------------------
IpAddress                           40
Counter (Counter32 in SNMPv2)       41
Gauge (Gauge32 in SNMPv 2)          42
TimeTicks                           43
Opaque                              44
NsapAddress                         45
Counter64 (available only in SNMPv2)  46
```

```
Uinteger32 (available only in SNMPv2) 47

Context-specific types within an SNMP Message Identifier in hex
---------------------------------------------------------------
GetRequest-PDU                            A0
GetNextRequestPUD                         A1
GetResponse-PDU (Response-PDU in SNMPv 2) A2
SetRequest-PDU                            A3
Trap-PDU (obsolete in SNMPv 2)            A4
GetBulkRequest-PDU (added in SNMPv 2)     A5
InformRequest-PDU (added in SNMPv 2)      A6
SNMPv2-Trap-PDU (added in SNMPv 2)        A7
*/
```

If you want to send more information in the trap, you can use the timestamp field or the specific trap field. If you are using NETVIEW you will see the message on the z/OS system that receives the trap.

```
SNM030I SNMP request 1001 received following trap:
SNM031I Agent Address: 164.25.130.1
SNM032I Generic trap type: 4
SNM033I Specific trap type: 0
SNM034I Time stamp: 305419896
SNM035I Enterprise Object ID: 1.3.6.1.4.1.2.3.13
SNM039I SNMP request 1001 End of trap data
```

3.6 Monitoring performance using MIB information and traps

The SLA Performance Monitor MIB (SLAPM-MIB) subagent provides monitoring through PolicyName, PolicyRuleStats, PRMon, and Subcomponent tables. The SLA Subagent communicates with the SNMP agent to provide GET and SET support for various MIB tables and objects, and also trap generation. Traps can be generated for various performance out-of-bounds conditions, for example, maximum delay exceeded, and for significant events such as policy deletion.

3.6.1 SLAPM-MIB and traps monitored by the SLA Subagent

The SLA Subagent provides information about service policies and performance data about applications mapped to those policies by using two tables. See Table 3-8.

Table 3-8 SLAPM-MIB monitoring tables

SLAPM-MIB table name	Description
slapmPolicyNameTable	Maps an index to a unique name associated with a given policy rule definition.
slapmPolicyRuleStatsTable	Provides information about defined service policies and aggregate performance data for mapped applications. This table is functionally equivalent to the deprecated slapmPolicyStatsTable.
slapmSubcomponentTable	Provides information about individual TCP or UDP applications and application-specific performance data.

The SLA Subagent also supports performance monitoring through the slapmPRMonTable object. This table is functionally equivalent to the deprecated slapmPolicyMonitorTable. Entries are created in the monitor table to establish the desired criteria for monitoring. The following levels of monitoring are provided:

- Aggregate

  Monitoring is performed based on the aggregate of all TCP or UDP applications that are mapped to one or more service policies.

- Subcomponent

  Monitoring is performed based on a single TCP or UDP application.

Three types of monitoring are provided for measuring application performance. They are described in Table 3-9.

Table 3-9 Monitoring types

Monitoring type	Description
MinRate	The current input/output rates of the application(s) are compared to threshold values established in the monitor table entry. If the current rates are less than the threshold, an SNMP trap is sent if traps are enabled.
MaxRate	The current input/output rates of the application(s) are compared to threshold values established in the monitor table entry. If the current rates are greater than the threshold, an SNMP trap is sent if traps are enabled.
MaxDelay	The current delay rates of the application(s) are calculated by using TCP round trip time (RTT). For aggregate monitoring, the RTT of all TCP applications are averaged. The delay rates are compared to threshold values established in the monitor table entry. If the current rates are greater than the threshold, an SNMP trap is sent if traps are enabled.

Note: MaxDelay monitoring is only available for TCP applications.

You will see the detailed definition of the SLA MIB in the draft RFC, which is shipped with CS for z/OS 1.2 as /usr/lpp/tcpip/samples/slapm.txt.

The following four sets of SLAPM-MIB objects are supported by the SLA Subagent:

- Scalar objects
- Policy Stats table
- Policy Monitor table
- Subcomponent table

Several MIB objects are used in establishing monitor table entries and for configuring whether and how often traps are sent. Table 3-10 on page 58 shows significant MIB objects to monitor the performance and/or control the generation of traps.

Table 3-10 SLAPM-MIB objects

MIB object	Action
slapmPRMonControl	Controls what levels and types of monitoring and/or traps are in effect.
slapmPRMonInterval	Sets the interval for calculating input/output and delay rates and checking those values against the monitor table thresholds.
slapmPRMonMinRateLow	Establishes the lower threshold values for MinRate in units of kilobits per second.
slapmPRMonMinRateHigh	Establishes the upper threshold values for MinRate in units of kilobits per second.
slapmPRMonMaxRateLow	Establishes the lower threshold values for MaxRate in units of kilobits per second.
slapmPRMonMaxRateHigh	Establishes the upper threshold values for MaxRate in units of kilobits per second.
slapmPRMonMaxDelayLow	Establishes the lower threshold values for MaxDelay in units of milliseconds.
slapmPRMonMaxDelayHigh	Establishes the upper threshold values for MaxDelay in units of milliseconds.
slapmPRMonRowStatus	The value of this object indicates whether the row is active, in the process of being created, or temporarily not in use.
slapmPolicyTapEnable	Enables or suppresses the generation of PolicyDeleted and MonitorDeleted traps.
slapmPolicyTrapFilter	Establishes the number of times a given MinRate, MaxRate, or MaxDelay event must be encountered before a trap is generated.
slapmPolicyPurgeTime	Specifies a timeout value for Policy Deleted traps.

Note: For CS for z/OS 1.2, the SLA Subagent objects are experimental and occupy the experimental portion of the MIB tree. The MIB objects will change in future releases.

When SNMP traps are enabled, a *not achieved* trap is sent, as described in Table 3-9 on page 57. A corresponding *okay* trap is sent when the traffic once again conforms to the boundaries established in the monitor table entry. See Figure 3-13 on page 59 for an overview of performance monitoring.

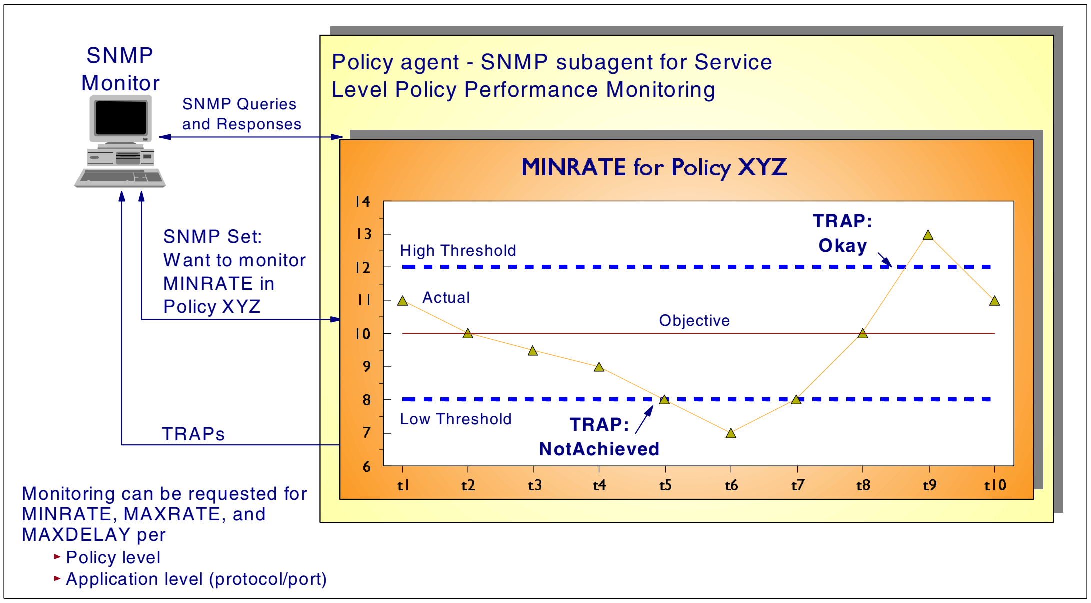

SNMP Monitor
SNMP Queries and Responses
SNMP Set: Want to monitor MINRATE in Policy XYZ
TRAPs
Monitoring can be requested for MINRATE, MAXRATE, and MAXDELAY per
Policy level
Application level (protocol/port)
Policy agent - SNMP subagent for Service Level Policy Performance Monitoring
MINRATE for Policy XYZ
14
13
12
11
10
9
8
7
6
High Threshold
Actual
Objective
Low Threshold
TRAP: Okay
TRAP: NotAchieved
t1
t2
t3
t4
t5
t6
t7
t8
t9
t10

Figure 3-13 Service level agreement performance monitoring

In addition to the traps used to measure application performance, two additional traps are used to monitor table administration, the PolicyDeleted and MonitorDeleted traps.

The detailed information on the monitoring traps is shown in the table below. These traps are defined as part of the IBM Enterprise-Specific trap. All enterprise-specific traps are generated with a trap value of 6.

Table 3-11 SNMP TCP/IP subagent enterprise-specific trap types: SLAPM traps

Value	Type	Description
7	slapmPolicyRuleMonNotOkay	This notification is generated when a monitored event is not achieved with respect to a threshold. This applies only to monitoring a policy rule as an aggregate via an associating slapmPolicyRuleStatsEntry. The value of slapmPRMonControl can be examined to determine what is being monitored. The first slapmPRMonStatus value supplies the current monitor status while the second value supplies the previous status.

Value	Type	Description
8	slapmPolicyRuleMonOkay	This notification is generated when a monitored event has improved to an acceptable level. This applies only towards monitoring a policy rule as an aggregate via an associating slapmPolicyRuleStatsEntry. The value of slapmPRMonControl can be examined to determine what is being monitored. The first slapmPRMonStatus value supplies the current monitor status while the second value supplies the previous status.
9	slapmPolicyRuleDeleted	A slapmPolicyRuleDeleted notification is sent when a slapmPolicyRuleStatsEntry is deleted if the value of slapmPolicyTrapEnable is enabled(1).
10	slapmPolicyRuleMonDeleted	A slapmPolicyRuleMonDeleted notification is sent when a slapmPRMonEntry is deleted if the value of slapmPolicyTrapEnable is enabled(1).
11	slapmSubCMonitorNotOkay	This notification is generated when a monitored value does not achieved a threshold specification. This applies only towards monitoring the individual components of a policy rule. The value of the corresponding slapmPRMonControl can be examined to determine what is being monitored. The first slapmSubcomponentMonitorStatus value supplies the current monitor status while the second value supplies the previous status.
12	slapmSubCMonitorOkay	This notification is generated when a monitored value has reached an acceptable level.

3.6.2 Extracting the performance information from SLAPM-MIB

To get performance information from SLAPM-MIB, follow the steps below.

Step 1: Start the SNMP agent, the Policy Agent, and the SLA Subagent

If the policy information has been installed in the TCP/IP stack, the SLA Subagent creates the following objects automatically:

- Scalar objects
- Policy name table
- Policy status table
- Subcomponent table

Note: If you do not have the policy information installed, only the global objects are created.

At ITSO Raleigh we used the z/OS LDAP Server to provide the policy information to the policy agent. Example 3-8 shows some of our policy rule definitions. We use the same policies for system RA28 and RA39.

Example 3-8 Policy definitions at ITSO Raleigh

```
dn:pr=ftprule, pg=TCPIPC, g=policy, o=IBM_US, c=US
objectclass:ibm-policyRule
ibm-policyRuleName:ftprule
cn:default application - ftprule
ibm-policyRuleEnabled:1
ibm-policyRuleConditionListType:2
ibm-policyRuleConditionList:1:+:pc=cond1, pg=TCPIPC, g=policy, o=IBM_US, c=US
ibm-policyRuleActionList:255:pa=ftpaction, pg=TCPIPC, g=policy, o=IBM_US, c=US
ibm-policyRuleValidityPeriodList:pc=period9, pg=TCPIP, g=policy, o=IBM_US, c=US
ibm-policyRuleKeywords:testingPolicyRules
ibm-policyRulePriority:2
ibm-policyRuleMandatory:TRUE
ibm-policyRuleSequencedActions:1

dn:pr=ospf, pg=TCPIP, g=policy, o=IBM_US, c=US
objectclass:ibm-policyRule
ibm-policyRuleName:ospf
cn:default application - ospf
ibm-policyRuleEnabled:1
ibm-policyRuleConditionListType:2
ibm-policyRuleConditionList:1:+:pc=cond2,  pg=TCPIP, g=policy, o=IBM_US, c=us
ibm-policyRuleActionList:1:pa=networkcontrol, pg=TCPIP, g=policy, o=IBM_US, c=US
ibm-policyRuleValidityPeriodList:pc=period9, pg=TCPIP, g=policy, o=IBM_US, c=us
ibm-policyRuleKeywords:testingPolicyRules
ibm-policyRuleMandatory:TRUE
ibm-policyRuleSequencedActions:2
```

Step 2: Get appropriate index values

The SLA Subagent uses new table structures to map the policy rules. A new slapmPolicyNameTable has been defined and organized in five general classes:

- policyGroup
- policyRule
- policyCondition
- policyTimePeriodCondition
- policyAction

These five classes map the policy and group definition in an LDAP Server, and are reported in a slapmPolicyNameTable arranged with an arbitrary index. There are the same number of entries in the table as policies you have enabled in your TCP/IP stack.

There is one entry in PolicyNameTable for each policy enabled in your TCP/IP stack, and the value of the NameOfRule object is either the distinguished name of the policy if it has been defined in the LDAP Server or the policy name if it is defined in the PAGENT file.

Figure 3-14 illustrates the relationship between the NameTable entry and the policy rule definition.

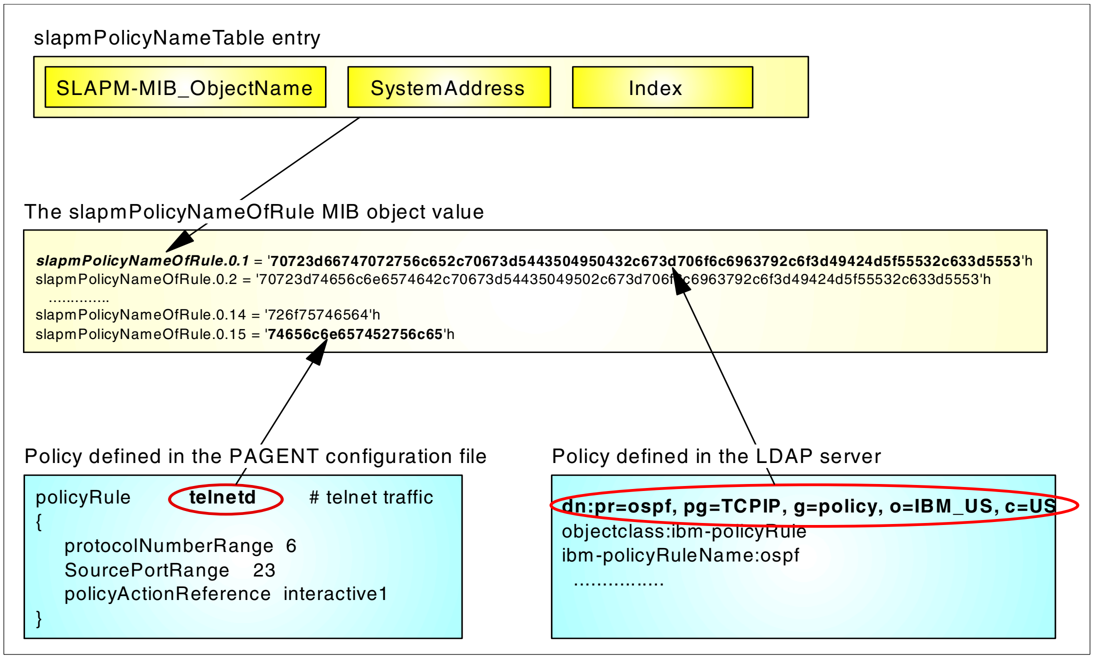

slapmPolicyNameTable entry
SLAPM-MIB_ObjectName
SystemAddress
Index
The slapmPolicyNameOfRule MIB object value
slapmPolicyNameOfRule.0.1 = '70723d66747072756c652c70673d5443504950432c673d706f6c6963792c6f3d49424d5f55532c633d5553'h
slapmPolicyNameOfRule.0.2 = '70723d74656c6e6574642c70673d54435049502c673d706f6c6963792c6f3d49424d5f55532c633d5553'h
....
slapmPolicyNameOfRule.0.14 = '726f75746564'h
slapmPolicyNameOfRule.0.15 = '74656c6e657452756c65'h
Policy defined in the PAGENT configuration file
policyRule telnetd # telnet traffic
{
protocolNumberRange 6
SourcePortRange 23
policyActionReference interactive1
}
Policy defined in the LDAP server
dn:pr=ospf, pg=TCPIP, g=policy, o=IBM_US, c=US
objectclass:ibm-policyRule
ibm-policyRuleName:ospf
....

Figure 3-14 slapmPolicyNameTable

In the example shown in Figure 3-15, the policy rules are stored in the LDAP Server. The policy agent has retrieved this information from the LDAP Server and installed it into a TCP/IP stack. In this case, the slapmPolicyNameOfRule MIB value is identical to the distinguished name defined for the corresponding LDAP object.

```
PEDRAT @ RA28:/etc>osnmp -v -h mvs28a walk slapmPolicyNameTable

slapmPolicyNameOfRule.0.1 =
'70723d66747072756c652c70673d5443504950432c673d706f6c6963792c6f3d49424d5f55532c633d5553'
h
slapmPolicyNameOfRule.0.2 =
'70723d74656c6e6574642c70673d54435049502c673d706f6c6963792c6f3d49424d5f55532c633d5553'h
slapmPolicyNameOfRule.0.3 =
'70723d667470642c70673d54435049502c673d706f6c6963792c6f3d49424d5f55532c633d5553'h
slapmPolicyNameOfRule.0.4 =
'70723d6f7370662c70673d54435049502c673d706f6c6963792c6f3d49424d5f55532c633d5553'h
```

Figure 3-15 slapmPolicyNameTable - Policy in the LDAP Server

Figure 3-16 on page 63 is an example where the policy is defined in the PAGENT configuration file. In this scenario, the policy rule name will be used for the slapmPolicyNameOfRule MIB object value.

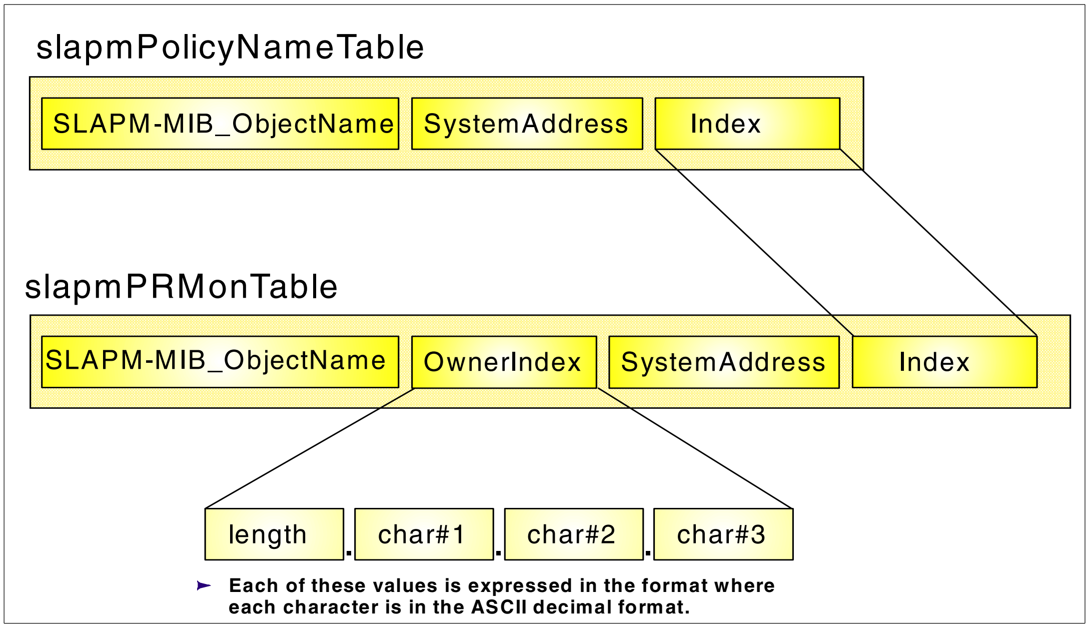
```
PEDRAT @ RA28:/etc>osnmp -v -h mvs28a walk slapmPolicyNameTable

slapmPolicyNameOfRule.0.1 = '525356505f52756c6531'h
slapmPolicyNameOfRule.0.2 = '45452d6c6f77707269'h
slapmPolicyNameOfRule.0.3 = '45452d6d6564707269'h
slapmPolicyNameOfRule.0.4 = '45452d68696768707269'h
slapmPolicyNameOfRule.0.5 = '45452d6e6574776f726b'h
slapmPolicyNameOfRule.0.6 = '45452d786964'h
slapmPolicyNameOfRule.0.7 = '646e732d746370'h
slapmPolicyNameOfRule.0.8 = '646e732d756470'h
slapmPolicyNameOfRule.0.9 = '7765622d6874747064'h
slapmPolicyNameOfRule.0.10 = '74656c6e657464'h
slapmPolicyNameOfRule.0.11 = '66747064'h
slapmPolicyNameOfRule.0.12 = '7466747064'h
slapmPolicyNameOfRule.0.13 = '6f737066'h
slapmPolicyNameOfRule.0.14 = '726f75746564'h
slapmPolicyNameOfRule.0.15 = '74656c6e657452756c65'h
```

Figure 3-16 slapmPolicyNameTable - Policy in PAGENT file

To monitor the performance with the SLA Subagent, the monitor table must be explicitly created using SNMP SET commands. When you create monitor table entries, the appropriate index value has to be specified. The index is composed of the following object values:

- slapmPRMonOwnerIndex
- slapmPRMonSystemAddress
- slapmPRMonIndex

The slapmPRMonIndex is used to index the slapmPolicyName table, which contains the policy rules. See Figure 3-17 for the structure of the index and the relationship between entries in slapmPolicyNameTable and slapmPRMonTable.



➢ Each of these values is expressed in the format where each character is in the ASCII decimal format.

Figure 3-17 Table index structure

To create your index, you have to define the OwnerIndex, which is an ASCII character string represented in the hexdecimal format. For example, the value u1 is expressed as 2.117.49.

Now you have to find the index value that matches the policy that you want to monitor. To obtain the index values in your SLAPM-MIB, you can walks the slapmPolicyNameTable (see Figure 3-15 on page 62). Note that the index value used in the slapmPolicyNameTable entries consists of the last two values used in the PolicyName table index, namely SystemAddress, and Index.

Then you can cut and paste the index value from the slapmPolicyNameTable entries, and add an OwnerIndex of your choosing at the beginning of the index. For the example shown in Figure 3-14 on page 62, the index value from the PolicyName table is 0.1 for the policy rule ospf (dn:pr=ospf, pg=TCPIP, g=policy, o=IBM_US, c=US) and 0.15 for the policy rule telnetd.

Since we chose the owner index of u1, the complete indexes will be:

```
2.117.49.0.1          for ospf
2.117.49.0.15         for telnetd
```

When you have created the slapmPRMonTable, the Index value refers to the entry in slapmPolicyNameTable to find the policy name and in the slapmPolicyRuleStatsTable to find status and information related to the same indexed policy. See Figure 3-18 on page 65 for the PolicyRuleStatus table.

```
PEDRAT @ RA28:/etc>osnmp -v -h mvs28a walk slapmPolicyRuleStatsTable

slapmPolicyRuleStatsOperStatus.0.1 = 2
slapmPolicyRuleStatsOperStatus.0.2 = 2
slapmPolicyRuleStatsActiveConns.0.1 = 0
slapmPolicyRuleStatsActiveConns.0.2 = 1
slapmPolicyRuleStatsTotalConns.0.1 = 0
slapmPolicyRuleStatsTotalConns.0.2 = 2
slapmPolicyRuleStatsLActivated.0.1 = 2000-7-18,13:23:44.0
slapmPolicyRuleStatsLActivated.0.2 = 2000-7-18,13:23:44.0
slapmPolicyRuleStatsLastMapping.0.1 = 0-0-0,0:0:0.0
slapmPolicyRuleStatsLastMapping.0.2 = 2000-7-18,15:39:7.8
slapmPolicyRuleStatsInOctets.0.1 = 0
slapmPolicyRuleStatsInOctets.0.2 = 311
slapmPolicyRuleStatsOutOctets.0.1 = 0
slapmPolicyRuleStatsOutOctets.0.2 = 345
slapmPolicyRuleStatsConnLimit.0.1 = 5
slapmPolicyRuleStatsConnLimit.0.2 = 0
slapmPolicyRuleStatsCountAccepts.0.1 = 0
slapmPolicyRuleStatsCountAccepts.0.2 = 2
slapmPolicyRuleStatsCountDenies.0.1 = 0
slapmPolicyRuleStatsCountDenies.0.2 = 0
slapmPolicyRuleStatsInDiscards.0.1 = 0
slapmPolicyRuleStatsInDiscards.0.2 = 0
slapmPolicyRuleStatsOutDiscards.0.1 = 0
slapmPolicyRuleStatsOutDiscards.0.2 = 0
slapmPolicyRuleStatsInPackets.0.1 = 0
slapmPolicyRuleStatsInPackets.0.2 = 13
slapmPolicyRuleStatsOutPackets.0.1 = 0
slapmPolicyRuleStatsOutPackets.0.2 = 14
slapmPolicyRuleStatsInProOctets.0.1 = 0
slapmPolicyRuleStatsInProOctets.0.2 = 0
slapmPolicyRuleStatsOutProOctets.0.1 = 0
slapmPolicyRuleStatsOutProOctets.0.2 = 0
slapmPolicyRuleStatsMinRate.0.1 = 0
slapmPolicyRuleStatsMinRate.0.2 = 0
slapmPolicyRuleStatsMaxRate.0.1 = 400
slapmPolicyRuleStatsMaxRate.0.2 = 0
slapmPolicyRuleStatsMaxDelay.0.1 = 0
slapmPolicyRuleStatsMaxDelay.0.2 = 0
slapmPolicyRuleStatsTotalRsvpFlows.0.1 = 0
slapmPolicyRuleStatsTotalRsvpFlows.0.2 = 0
slapmPolicyRuleStatsActRsvpFlows.0.1 = 0
slapmPolicyRuleStatsActRsvpFlows.0.2 = 0
```

Figure 3-18 Walking the slapmPolicyRuleStatsTable

Step 3: Create the monitor table objects

Now you are ready to create the monitor table objects. The table can be created by issuing several SET commands as shown in Figure 3-19 on page 66.

```
PEDRAT @ RA28:/etc>osnmp -v -h mvs28a set slapmPolicyTrapEnable.0 1      1
slapmPolicyTrapEnable.0 = 1
PEDRAT @ RA28:/etc>osnmp -v -h mvs28a set slapmPolicyTrapFilter.0 1      2
slapmPolicyTrapFilter.0 = 1
PEDRAT @ RA28:/etc>osnmp -v -h mvs28a set slapmPolicypurgeTime.0 60      3
slapmPolicyPurgeTime.0 = 60
PEDRAT @ RA28:/etc>osnmp -v -h mvs28a set slapmPRMonControl.2.117.49.0.1 \'0000003f\'h
slapmPRMonControl.2.117.49.0.1 = '0000003f'h                             4
PEDRAT @ RA28:/etc>osnmp -v -h mvs28a set slapmPRMonMinRateLow.2.117.49.0.1 10
slapmPRMonMinRateLow.2.117.49.0.1 = 10
PEDRAT @ RA28:/etc>osnmp -v -h mvs28a set slapmPRMonMinRateHigh.2.117.49.0.1 50
slapmPRMonMinRateHigh.2.117.49.0.1 = 50
PEDRAT @ RA28:/etc>osnmp -v -h mvs28a set slapmPRMonMaxRateLow.2.117.49.0.1 300
slapmPRMonMaxRateLow.2.117.49.0.1 = 300
PEDRAT @ RA28:/etc>osnmp -v -h mvs28a set slapmPRMonMaxRateHigh.2.117.49.0.1 350
slapmPRMonMaxRateHigh.2.117.49.0.1 = 350
PEDRAT @ RA28:/etc>osnmp -v -h mvs28a set slapmPRMonMaxDelayLow.2.117.49.0.1 50
slapmPRMonMaxDelayLow.2.117.49.0.1 = 50
PEDRAT @ RA28:/etc>osnmp -v -h mvs28a set slapmPRMonMaxDelayHigh.2.117.49.0.1 100
slapmPRMonMaxDelayHigh.2.117.49.0.1 = 100
PEDRAT @ RA28:/etc>osnmp -v -h mvs28a set slapmPRMonRowStatus.2.117.49.0.1 1
slapmPRMonRowStatus.2.117.49.0.1 = 1                                    5
```

Figure 3-19 Creating monitor table objects

Turn on the generation of PolicyDeleted and MonitorDeleted traps. 1

Setting this value to 1, each time a given MinRate, MaxRate or MaxDelay event is occurred, a trap is generated 2.

After 60 seconds a Service Policy is deleted, and a PolicyDeleted trap is generated 3.

Setting the slapmPRMonControl to x'3f' enables it to monitor all events and generate all kinds of SLAPM traps. After issuing this command, you will be able to walk through the slapmPRMonControl table 4.

The slapmPRMonRowStatus object controls the status of a monitor entry. To activate the entry, set the value to 1 5.

Note: To change any of the monitor table object values for an existing table entry or row, you must take the row out of service to make the changes. To do this, set the value of slapmPRMonRowStatus to 2.

Then you can walk through the monitor table entries. See Figure 3-20 on page 67.

```
PEDRAT @ RA28:/etc>osnmp -v -h mvs28a walk slapmPRMonTable

slapmPRMonControl.2.117.49.0.1 = '0000003f'h
slapmPRMonStatus.2.117.49.0.1 = '00000000'h
slapmPRMonInterval.2.117.49.0.1 = 20
slapmPRMonIntTime.2.117.49.0.1 = 2000-8-23,14:39:27.8
slapmPRMonCurrentInRate.2.117.49.0.1 = 0
slapmPRMonCurrentOutRate.2.117.49.0.1 = 0
slapmPRMonMinRateLow.2.117.49.0.1 = 10
slapmPRMonMinRateHigh.2.117.49.0.1 = 50
slapmPRMonMaxRateHigh.2.117.49.0.1 = 350
slapmPRMonMaxRateLow.2.117.49.0.1 = 300
slapmPRMonMaxDelayHigh.2.117.49.0.1 = 100
slapmPRMonMaxDelayLow.2.117.49.0.1 = 50
slapmPRMonMinInRateNotAchieves.2.117.49.0.1 = 0
slapmPRMonMaxInRateExceeds.2.117.49.0.1 = 0
slapmPRMonMaxDelayExceeds.2.117.49.0.1 = 0
slapmPRMonMinOutRateNotAchieves.2.117.49.0.1 = 0
slapmPRMonMaxOutRateExceeds.2.117.49.0.1 = 0
slapmPRMonCurrentDelayRate.2.117.49.0.1 = 0
slapmPRMonRowStatus.2.117.49.0.1 = 1
```

Figure 3-20 Objects in monitor table

Note: Note that every time you start the SLA Subagent, the monitor table has to be created by issuing the commands shown in Figure 3-20.

For easy operation, we used the following shell program to initialize and create the necessary table entries:

```
#!/bin/sh

DEST=$1
TMPFILE=./"tmp.""$$"
OWNER_INDEX="2.117.49"   # Owner index is set to "u1"

if [ "$DEST" = "" ]
then
        echo " Usage : $0 dest_SNMP_AgentID "
        exit 1
fi

#   Disable AuthenticationFailure Traps
SNMP_CMD="osnmp -h $DEST set snmpEnableAuthenTraps.0 2"
echo $SNMP_CMD
$SNMP_CMD
if [ $? != 0 ]
then
   exit 255
fi

#   Enable PolicyDeleted and MonitorDeleted Traps
SNMP_CMD="osnmp -h $DEST set slapmPolicyTrapEnable.0 1"
echo $SNMP_CMD
$SNMP_CMD
if [ $? != 0 ]
then
   exit 255
fi
```

```
SNMP_CMD="osnmp -h $DEST set slapmPolicyTrapFilter.0 1"
echo $SNMP_CMD
$SNMP_CMD
if [ $? != 0 ]
then
   exit 255
fi

SNMP_CMD="osnmp -h $DEST set slapmPolicyPurgeTime.0 60"
echo $SNMP_CMD
$SNMP_CMD
if [ $? != 0 ]
then
   exit 255
fi

#   Retrieve part of the index from PolicyNameTable
SNMP_CMD="osnmp -v -h $DEST walk slapmPolicyNameTable"
$SNMP_CMD | grep slapmPolicyNameOfRule | cut -c23-999 > $TMPFILE
 if [ $? != 0 ]
 then
   exit 255
 fi


#   While loop to support multiple policies installed.
cat $TMPFILE |
while read line
do
     set $line
     INDEX="$OWNER_INDEX.$1"
     echo $INDEX

     SNMP_CMD="osnmp -h $DEST set slapmPRMonRowStatus.$INDEX 2"
     echo $SNMP_CMD
     $SNMP_CMD

     SNMP_CMD="osnmp -h $DEST set slapmPRMonControl.$INDEX '0000003F'h"
     echo $SNMP_CMD
     $SNMP_CMD
     if [ $? != 0 ]
     then
        exit 255
     fi

     SNMP_CMD="osnmp -h $DEST set slapmPRMonRowStatus.$INDEX 1"
     echo $SNMP_CMD
     $SNMP_CMD
     if [ $? != 0 ]
     then
        exit 255
     fi
done

#   Display the PolicyMonitorTable objects.
echo ""
SNMP_CMD="osnmp -v -h $DEST walk slapmPRMonTable"
echo $SNMP_CMD
$SNMP_CMD
```

```
/bin/rm $TMPFILE
echo "Table initialization completed."
exit 0
```

This shell program requires one parameter, the winSNMPName parameter configured in your /etc/osnmp.conf file. Using this shell program, you will create and initialize each monitor table for all policies installed in the TCP/IP stack. The MIB setting could be changed to meet your monitor policy.

Two of the monitor table objects, fields PRMonControl and PRMonStatus, are important in setting up the entries and understanding why traps are generated. Both of these fields have the SNMP data type BITS, which means they are bit strings, where bit 0 is the low order bit. Any combination of bits can be set into these objects. Refer to Table 3-12 and Table 3-13 for the possible values.

Table 3-12 slapmPRMonControl objects values

slapmPRMonControl	Bits value (xx54 3210)
0 - monitor MinRate	0000 0001 = x’01’
1 - monitor MaxRate	0000 0010 = x’02’
2 - monitor MaxDelay	0000 0100 = x’04’
3- enable aggregate traps	0000 1000 = x’08’
4 - enable subcomponent traps	0001 0000 = x’10’
5 - monitor subcomponent	0010 0000 = x’20’

Table 3-13 slapmPRMonStatus objects values

slapmPRMonStatus	Bits value (xx98 7654 3210)
0 - slaMinInRateNotAchieved	0000 0000 0001 = x’001’
1 - slaMaxInRateExeeded	0000 0000 0010 = x’002’
2 - slaMaxDelayExceeded	0000 0000 0100 = x’004’
3 - slaMinOutRateNotAchieved	0000 0000 1000 = x’008’
4 - slaMaxOutRateExceeded	0000 0001 0000 = x’010’
5 - monitorMinInRateNotAchieved	0000 0010 0000 = x’020’
6 - monitorMaxInRateExceeded	0000 0100 0000 = x’040’
7 - monitorMaxDelayExceeded	0000 1000 0001 = x’080’
8 - monitorMinOutRateNotAchieved	0001 0000 0000 = x’100’
9 - monitorMaxOutRateExceeded	0010 0000 0000 =x’200’

As you can see in Table 3-12, by extracting the values of the slapmPRMonStatus objects using the GET SNMP command, you can get the current status of the monitored traffic.

For more information about performance monitoring, please refer to:

- *z/OS V1R2 Communications Server: IP Configuration Reference*, SC31-8776
- *z/OS V1R2 Communications Server: IP Configuration Guide*, SC31-8775
- *Managing OS/390 TCP/IP with SNMP*, SG24-5866

4

Working with z/OS Policy Agent

On CS for z/OS 1.2 the policy system provided by the Communication Server component is called the Policy Agent (PAGENT). The repository where the policies are stored could be either PAGENT's configuration file or in an LDAP Server.

CS for z/OS 1.2 provides the Policy Agent that serves as a central point of control/access for all policies. The policies supported by the Policy Agent can be used for any of the following functions:

- Differentiated Services
- Integrated Services
- Sysplex Distributor
- Traffic Regulation and Management
- Scan detection
- Attack detection

In addition, the Policy Agent supports functions other than reading and installing policies, such as Sysplex Distributor policy performance monitoring and mapping Type of Service (TOS) byte values to outbound interface priorities.

4.1 Implementing PAGENT on z/OS

On CS for z/OS 1.2 the Policy Agent runs as a UNIX process. As such, it may be started either from the UNIX System Services shell or as a started task. At ITSO Raleigh we used a started task procedure to start Policy Agent.

4.1.1 Starting PAGENT as started task

The following is a Policy Agent started task procedure that we used in our system:

```
//PAGENT   PROC
//*
//* SecureWay Communications Server IP
//* SMP/E distribution name: EZAPAGSP
//*
//* 5647-A01 (C) Copyright IBM Corp. 1999.
//* Licensed Materials - Property of IBM
//*
//PAGENT   EXEC PGM=PAGENT,REGION=0K,TIME=NOLIMIT,
//      PARM='POSIX(ON) ALL31(ON) ENVAR("_CEE_ENVFILE=DD:STDENV")/'
//*
//*       PARM='POSIX(ON) ALL31(ON) ENVAR("_CEE_ENVFILE=DD:STDENV")/-c /
//*             etc/pagent.r2615.conf -d'                1
//*      PARM='POSIX(ON) ALL31(ON) ENVAR("_CEE_ENVFILE=DD:STDENV")/-d'
//*
//* Example of passing parameters to the program (parameters must
//* extend to column 71 and be continued in column 16):
//*
//*STDENV    DD DSN=TCPIP.TCPPARMS.R2615(PAG&SYSCLONE.ENV),DISP=SHR
//STDENV   DD PATH='/etc/pagent.pok.env',PATHOPTS=(ORDONLY)    2
//*
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//CEEDUMP  DD SYSOUT=*,DCB=(RECFM=FB,LRECL=132,BLKSIZE=132)
```

You can use environment variables, either configured in an MVS data set or HFS file specified by the STDENV DD 1 to run with the desired configuration. We have configured environment variables in an HFS file 2, /etc/pagent.pok.env which contains:

```
BROWSE -- /etc/pagent.pok.env --------------------
 Command ===>
********************************* Top of Data *******
LIBPATH=/lib:/usr/lib:/usr/lpp/ldapclient/lib:.   3
PAGENT_CONFIG_FILE=/etc/pagent.pok.conf           4
PAGENT_LOG_FILE=/tmp/pagent.pok.log               5
PAGENT_LOG_FILE_CONTROL=300,3                     6
_BPXK_SETIBMOPT_TRANSPORT=TCPIPB                  7
TZ=EST5EDT                                        8
******************************** Bottom of Data *****
```

We configured six environment variables for the Policy Agent. The first variable, LIBPATH, enables PAGENT to search the dynamic link libraries needed to act as an LDAP client 3. The PAGENT_CONFIG_FILE specifies the PAGENT configuration file to use 4. The PAGENT_LOG_FILE specifies the log file name used by PAGENT 5, and the PAGENT_LOG_FILE_CONTROL 6 defines how many PAGENT log files are used and the size of the files. These are used in a round-robin (circular) fashion. We used the default value.

To configure PAGENT to use SYSLOGD to log messages, this means you define PAGENT_LOG_FILE=SYSLOGD. In this case the PAGENT_LOG_FILE_CONTROL has no meaning.

In our case, while we do not have the RESOLVER_CONFIG variable configured, PAGENT establishes an affinity to the proper TCP/IP stack through BPXK_SETIBMOPT_TRANSPORT=TCPIPB (**7**). Alternatively, we could have used the TCP/IP image name configured in the TcpImage statement in the Policy Agent configuration file to determine to which TCP/IP it will install policies.

For the Policy Agent to run in your local time zone, you might have to specify the time zone in your working location using the TZ environment variable (**8**) even if you have the TZ environment variable configured in /etc/profile. Note that most z/OS UNIX applications that start as MVS started tasks cannot use environment variables that have been configured in /etc/resolve.conf.

4.1.2 Define security product authorization for PAGENT

Because the Policy Agent can affect system operation significantly, security product authority (for example, RACF) is required to start the Policy Agent from a z/OS procedure library.

First, define a user ID for PAGENT with a UID of 0 and define it to the RACF started class list.

```
ADDUSER PAGENT   DFLTGRP(OMVSGRP) OMVS(UID(0) HOME('/'))
RDEFINE STARTED  PAGENT.* STDATA(USER(PAGENT))
```

Then define MVS.SERVMGR.PAGENT to the OPERCMDS class and permit the user ID that PAGENT is running under to it.

```
RDEF OPERCMDS (MVS.SERVMGR.PAGENT) UACC(NONE)
PE MVS.SERVMGR.PAGENT CLASS(OPERCMDS) ACCESS(CONTROL) ID(userid)
SETR RACLIST(OPERCMDS) REFRESH
```

These sample commands can be found in the EZARACF sample in SEZAINST.

If Policy Agent clients (such as, pasearch) are *not* defined as a super user, then security product authority in the SERVAUTH CLASS for that client *must* be defined to retrieve policies. The format of the profile under the SERVEAUTH class is:

```
EZB.PAGENT.sysname.TcpImage.ptype
```

Where:

- *sysname* is the system name defined in sysplex.
- *TcpImage* is the TCP name for policy information being requested.
- *ptype* is the policy type that is being requested, where:
  - QOS is the Policy QoS.
  - IDS is the Policy IDS.

Note: Wildcarding is allowed on segments of the profile name.

To set this up the following RACF commands can be used:

```
SETROPTS CLASSACT(SERVAUTH)
SETROPTS RACLIST(SERVAUTH)
SETROPTS GENERIC(SERVAUTH)
RDEFINE SERVAUTH EZB.PAGENT.sysname.tcpprocname.* UACC(NONE)
PERMIT EZB.PAGENT.sysname.tcpprocname.* CLASS(SERVAUTH) -
  ID(userid) ACCESS(READ)
SETROPTS GENERIC(SERVAUTH) REFRESH
SETROPTS RACLIST(SERVAUTH) REFRESH
```

The Policy Agent will check all client's requests to verify SERVAUTH class is active and the profile exists for the TcpImage(s) and policy type(s) in the request. If a client's request is for *all* TcpImages and policy types defined, then the Policy Agent will only return information for any object for which permission is granted. For example, if the request is for *all* policy types, and both QoS and IDS policy types are defined but the user is only granted permission for the QoS policy types, then only QoS policy information will be returned.

If the SERVAUTH class is absent (not RACLISTed), or profiles are absent for a client's request (that is, TcpImage, policy type), then permission is denied and data is *not* returned.

If the SERVAUTH class is active and profiles are present for a client's request (that is, TcpImage and policy type) and an MVS user is defined for all profiles, then permission is granted and data is returned.

If the SERVAUTH class is active and profiles are present for a client's request (that is, TcpImage and policy type) and an MVS user (that is, Policy Agent client) is not defined for all profiles, then permission is refused and data is not returned.

4.1.3 Starting PAGENT from UNIX

The PAGENT executable resides in /usr/lpp/tcpip/sbin. There is also a link from /usr/sbin. Make sure your PATH statement contains either /usr/sbin or /usr/lpp/tcpip/sbin.

To start PAGENT in the z/OS UNIX System Services shell you simply need to issue:

```
pagent -c /etc.pagent.pok.conf
```

4.1.4 Basic configuration

Before defining policies, some basic operational characteristics of the Policy Agent need to be configured in the PAGENT configuration file. In this section, we detail the following configuration steps:

- Define the TcpImage statements.
- Define the appropriate logging level.

Define the TcpImage statement(s)

The Policy Agent can be configured to install policies on one or more TCP/IP stacks or images. Each TCP/IP stack is configured using a TcpImage statement in the main configuration file. A secondary configuration file can be defined for each stack, a set of stacks can share configuration information in the main configuration file, or a combination of these techniques can be used.

To install different sets of policies to different stacks, configure each image with a different secondary configuration file. In this case, each image can be configured with a different policy refresh interval if desired. The refresh interval used for the main configuration file will be the smallest of the values specified for the different stacks.

In Figure 4-1 on page 75 we show PAGENT’s configuration file identifying, through TcpImage statements, the names of the TCP/IP stacks that policies are to be installed and the different configuration files that should be used by each.

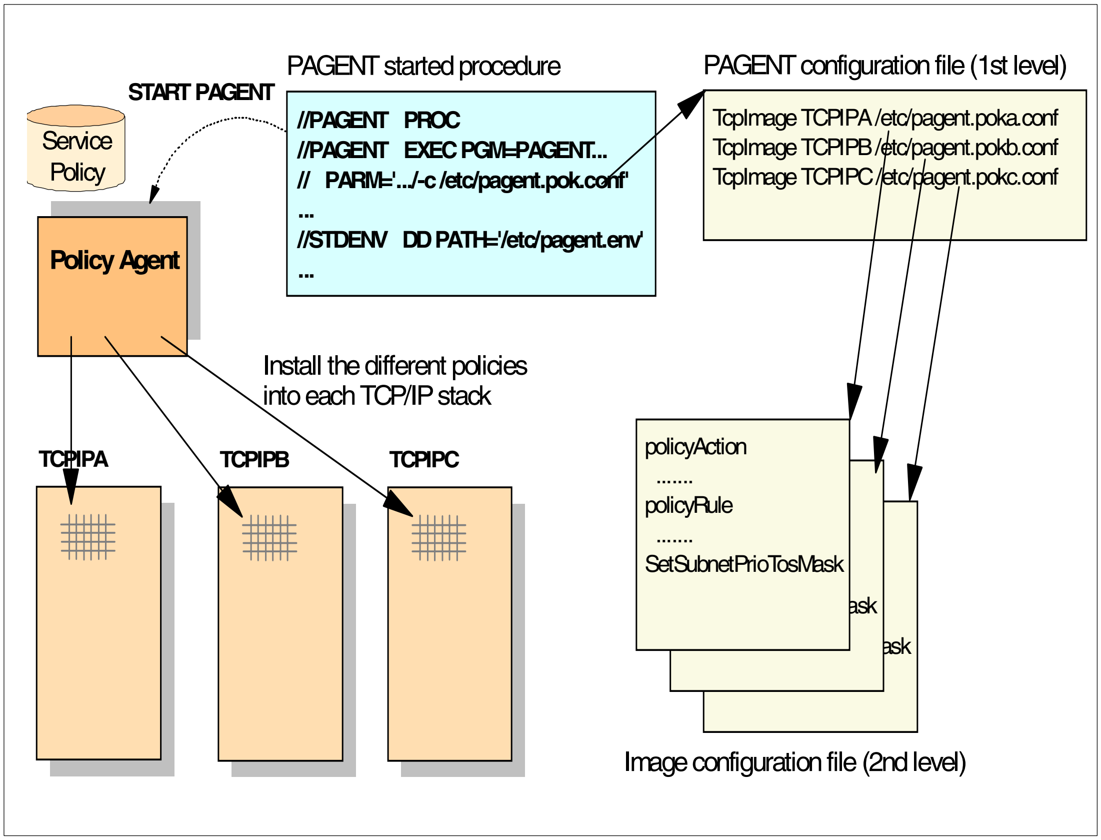

PAGENT started procedure
START PAGENT
Service Policy
//PAGENT PROC
//PAGENT EXEC PGM=PAGENT...
// PARM='.../-c /etc/pagent.pok.conf'
...
//STDENV DD PATH='/etc/pagent.env'
...
PAGENT configuration file (1st level)
TcpImage TCPIPA /etc/pagent.poka.conf
TcpImage TCPIPB /etc/pagent.pokb.conf
TcpImage TCPIPC /etc/pagent.pokc.conf
Policy Agent
Install the different policies
into each TCP/IP stack
TCPIPA
TCPIPB
TCPIPC
policyAction
....
policyRule
....
SetSubnetPrioTosMask
ask
ask
Image configuration file (2nd level)

Figure 4-1 Multiple stacks, multiple policy definitions

> **Note:** When the main configuration file is an MVS data set, it is reread at each refresh interval (which is the smallest of the individual stack refresh intervals), regardless of whether it has actually been changed or not. Because PAGENT restarts all stack-related processing when the main configuration file is reread, this effectively makes the refresh interval for all stacks the same as this smallest configured interval.

To install a common set of policies to a set of stacks, do not specify secondary configuration files for each image. In this case, there is only one configuration file (the main one) and the policy information contained in it is installed to all of the configured stacks. Different refresh intervals can also be configured for each image, but would probably be less useful in this case.

In Figure 4-2 on page 76 we show PAGENT's configuration file identifying, through TcpImage statements, the names of the TCP/IP stacks that policies are to be installed but in this case installing the same policies into each.

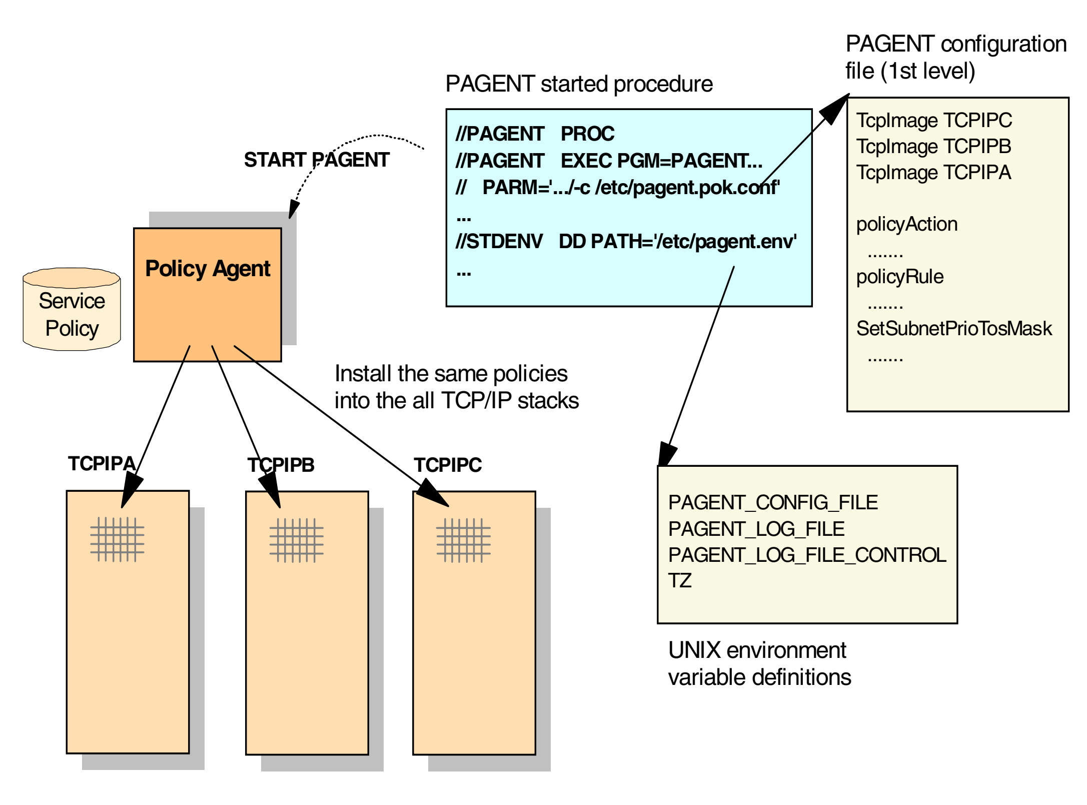


Figure 4-2 Multi-stack, single policy definition

In either case, it is possible that TCP/IP stacks configured to the Policy Agent are not started or even defined. The Policy Agent will fail when trying to connect to those stacks and log appropriate error messages.

The Policy Agent does not end when any (or all) stacks end. When the stack(s) are restarted, active policies are automatically reinstalled.

When the Policy Agent is shut down normally (that is, using KILL or STOP commands), then if the TcpImage statement option PURGE was coded, all policies will be purged from this stack.

The TcpImage statement specifies a TCP/IP image and its associated configuration file to be installed to that image. The following example installs the policy control file /tmp/TCPCS.policy to the TCPCS TCP/IP image, after flushing the existing policy control data:

```
TcpImage TCPCS /tmp/TCPCS.policy FLUSH
```

Define the appropriate logging level

The LogLevel statement is used to define the amount of information to be logged by the Policy Agent. The default is to log only event, error, console, and warning messages. This might be appropriate for a stable policy configuration, but more information might be required to understand policy processing or debug problems when first setting up policies or when making significant changes. Specify the LogLevel statement with the appropriate logging level in the main configuration file.

Note: The maximum logging level (511) can produce a significant amount of output, especially with large LDAP configurations. This is not of concern if HFS log files are used, as Policy Agent will round-robin (circulate) a set of finite size files. (The environmental variable PAGENT_LOG_FILE_CONTROL identifies the number and size of these files.) However, when using the syslog daemon as the log file, the amount of log output produced should be taken into consideration.

4.1.5 Considerations when defining policy rules

When you define and code the policy rules direction, source, and destination, you should consider when policy rules are applied.

- For TCP, the policies are applied at TCP connection set up.
- For UDP, a policy rule is applied every time a UDP datagram is being received or sent.
- For other protocols, such as ICMP, OSPF, etc., every time an IP datagram is being received or sent, the policy rules are applied.
- The policies are re-mapped when the policy definitions are being updated or refreshed. The rules will be re-mapped for every ACK segment in a TCP flow to adjust for time-of-day related policies.

4.2 Coding policy definitions in a configuration file

This example configuration is based upon Figure 4-1 on page 75. In this scenario, the policy definitions have been configured statically in the PAGENT configuration file and we use a two-level PAGENT configuration file to define the policy in a multiple IP stacks environment as shown below:

```
#  SecureWay Communications Server IP for OS/390
#  5647-A01 (C) Copyright IBM Corp. 1998, 1999.
#  Licensed Materials - Property of IBM
#
#      /etc/pagent.pok.conf  (PAGENT policy configuration)
#
# LogLevel Statement
Loglevel 255 1

# TcpImage Statement 2
TcpImage TCPIPA /etc/pagent.pokc.conf  FLUSH  A
TcpImage TCPIPB /etc/pagent.pokb.conf  FLUSH  B
TcpImage TCPIPC /etc/pagent.poka.conf  FLUSH  C
```

1 The log level is set with the integer that specified the level of logging/tracing. We are using LogLevel 255, which means all messages except trace messages are captured. The supported levels are:

- 1 - SYSERR - System error messages
- 2 - OBJERR - Object error messages
- 4 - PROTERR - Protocol error messages
- 8 - WARNING - Warning messages
- 16 - EVENT - Event messages
- 32 - ACTION - Action messages
- 64 - INFO - Informational messages
- 128 - ACNTING - Accounting messages
- 256 - TRACE - Trace messages

2 The TcpImage statement defined the TCP/IP stacks to be policed (A, B, and C). Up to four parameters can be configured for this statement. The first parameter specifies the TCP/IP stack name on which the policy must be installed. The next one is the path of the image configuration file for the associated TCP/IP stack.

For the third parameter, you can specify whether the Policy Agent deletes all the policies existing in the TCP/IP stack when it is started.

Note: The policies installed in the TCP/IP stack will be deleted at PAGENT startup time only if the FLUSH parameter is specified.

This prevents the policies from being deleted unexpectedly if PAGENT terminates abnormally. If you want to remove policies when you cancel PAGENT, you can restart PAGENT afterward, pointing to a configuration file with FLUSH specified but no policies defined.

In the last parameter, specify the time interval in seconds for checking the creation or modification time of the configuration file(s) and for refreshing policies from the LDAP Server. The default value is 1800 seconds (30 minutes).

Note: Dynamic monitoring for the configuration file is only supported for HFS files. MVS data sets are not monitored for changes.

The following shows the image configuration files for the TCP/IP stack TCPIPA defined in /etc/pagent.pok.conf. The same policies are defined for other stacks in different HFS files.

```
#  SecureWay Communications Server IP for OS/390
#  5647-A01 (C) Copyright IBM Corp. 1998, 1999.
#  Licensed Materials - Property of IBM
#
#       /etc/pagent.pok.conf  (PAGENT policy configuration)
#


# ReadFromDirectory  statement
#
# ReadFromDirectory       3
#{
#  LDAP_Server 172.16.250.3
#  LDAP_ProtocolVersion 3
#  LDAP_SchemaVersion 2
#  SearchPolicyBaseDN  pg=groupA, g=policy, o=IBM_US, c=US
# SearchPolicyBaseDN  pg=TCPIPC, g=policy, o=IBM_US, c=US
# SearchPolicyGroupKeyWord groupA
# SearchPolicyRuleKeyWord rule1
#}

# SetSubnetPrioTosMask  statement
SetSubnetPrioTosMask             4
{
SubnetAddr         0.0.0.0       A
SubnetTosMask      11100000 3    B
PriorityTosMapping 4 00000000    C
PriorityTosMapping 4 00100000
PriorityTosMapping 3 01000000
PriorityTosMapping 2 01100000
PriorityTosMapping 1 10000000
PriorityTosMapping 1 10100000
PriorityTosMapping 1 11000000
}

# PolicyAction statement
policyAction   networkcontrol    5
{
     policyScope    DataTraffic
     OutgoingTOS    11100000     # Precedence bits (first 3 bits)
}
policyAction   interactive1
{
     policyScope    DataTraffic
     OutgoingTOS    10000000
}
PolicyAction batch1
{
     policyScope    DataTraffic
     OutgoingTOS    01000000
}
PolicyAction Tscsrv      D
{
     policyScope    DataTraffic
     OutgoingTOS    01100000 E
      MaxRate          120
      Maxdelay       10
}

# PolicyRules statement       6

PolicyRules  ospf       5    # OSPF link advertisement traffic
{
     ProtocolNumber     89          # OSPF protocol number
     PolicyActionReference   networkcontrol
     PolicyName     ospf-mvs
}

PolicyRules  ftpd            # FTP traffic
{
     ProtocolNumber     TCP
     Direction          Both
     SourcePortRange    20 21       # Both FTP control and data ports
     PolicyActionReference   batch1   F
}

PolicyRules  telnetd         # telnet traffic
{
     ProtocolNumber     TCP
     Direction          Both
     SourcePortRange    23
     PolicyActionReference   interactive1
}

PolicyRules  TscsrvRule
{
     Direction           both
     PolicyScope         DataTraffic
     ProtocolNumber      TCP
     SourcePortRange     12000
     SourceAddressRange  172.16.232.1 172.16.252.254
     PolicyActionReference    Tscsrv
```

```
}
```

3 The ReadFromDirectory statement initializes the Policy Agent as an LDAP client. Therefore, if you use LDAP Servers to store the policy information, you must have this configured. For this statement, several parameters that identify the LDAP Server may be configured.

4 Define the TOS byte in the IP header using this statement.

The current IPv4 TOS byte format defines the first three bits to the precedence bits (for example, priority). Our default for the subnet TOS mask, if this statement is not specified, is 11100000. In this release, only QDIO devices in z/OS0 can support priorities. This statement is issued to set up TOS-to-priority mapping for those. QDIO supports four priority levels, 1 through 4, with 4 being the lowest priority.

In our sample configuration, we have configured:

A The local subnet interface address. 0.0.0.0 means the mask is the same for all interfaces.

B The TOS mask.

C Define the mapping between the TOS value and the priority level. This keyword can be repeated for each priority.

The default mappings are:

TOS	Priority
00000000	4
00100000	4
01000000	3
01100000	2
10000000	1
10100000	1
11000000	1
11100000	1

5 In the PolicyAction statement you can specify various parameters that identify the type of service for a flow, such as the name of the policy action (D) and the TOS value (E).

Note: Policies are applied to TCP on a connection basis, whereas they are applied to UDP/RAW on a per-packet basis.

The PolicyAction statement may also include:

- PolicyScope: Indicates to what traffic this policy rule applies. Valid values are DataTraffic, RSVP, Both (DataTraffic and RSVP), and TR.
- The maximum rate allowed for traffic in this service class.
- The minimum rate/throughput.
- The local transmission priority.
- Maximum end-to-end delay time.
- The maximum number of end-to-end connections at any instance of time.
- Days in a week that this Service Policy is active.
- The time of each day during which this Service Policy is active.
- The type of service being requested by RSVP applications: Controlled Load (the default) or Guaranteed.
- Parameters that limit the traffic rate for RSVP flows.
- Maximum number of reserved flows allowed for RSVP applications.
- Parameters for a TR policy and SD policy.

In our test we have mainly used the policy actions named networkcontrol, interactive1, batch1, and Tscsrv.

6 In the PolicyRules statement, you can define a group of IP datagrams that should receive a particular service. The TCP/IP applications specified in the PolicyRules are under the control of the Policy Agent according to the categories specified in the PolicyActionReference (F).

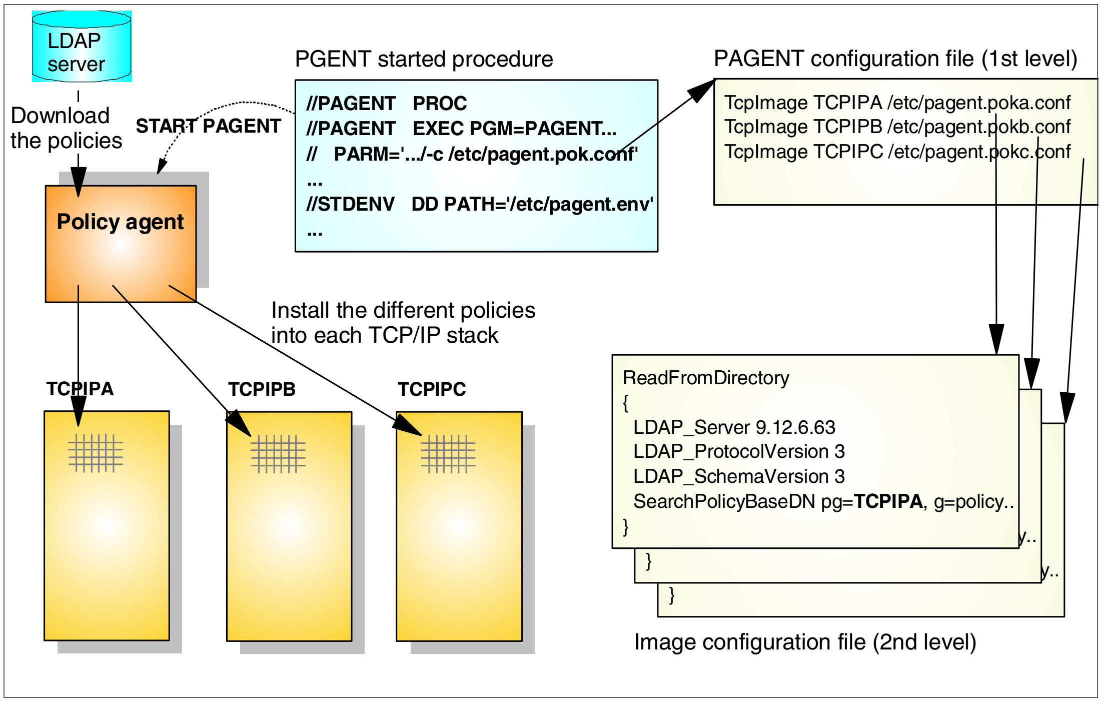
You may use the following parameters to identify the traffic or the scope in which this rule is applied.

- ProtocolNumber: Identifies the protocol running on top of IP with a one-byte field in the IP header.
- InboundInterface: The inbound local IP subnet to which this policy rule applies.
- OutboundInterface: The outbound local IP subnet to which this policy rule applies.
- SourceAddressRange: The local IP address range.
- DestinationAddressRange: The remote IP address range.
- SourcePortRange: The local port range.
- DestinationPortRange: The Remote Port range.
- DaysOfWeekMask: Specifies the days in a week that this policy rule is active.
- TimeOfDayRange: Indicates the time of day during which this policy rule is active.

4.3 PAGENT configuration file as an LDAP client

This section shows the same configuration that was used in 4.2, “Coding policy definitions in a configuration file” on page 77, except that now the policies are expressed in the format required for them to be installed in an LDAP Server.

Attention: Coding QoS and IDS policies in the format required for them to be stored in an LDAP Server is not a trivial task. QoS policies can either be coded and loaded from the TCP/IP configuration file or from and LDAP Server. IDS policies must be loaded from an LDAP Server. If you require your QoS and IDS policies to be stored in an LDAP Server, we at the ITSO strongly suggest that you use the zQoS Manager and zIDS Manager, which are explained in Chapter 6, “QoS configuration using the zQoS Manager” on page 125 and Chapter 10, “IDS configuration using zIDS Manager” on page 199, respectively.


LDAP server
Download the policies
START PAGENT
Policy agent
PGENT started procedure
PAGENT configuration file (1st level)
Install the different policies into each TCP/IP stack
TCPIPA
TCPIPB
TCPIPC
Image configuration file (2nd level)

```
//PAGENT   PROC
//PAGENT   EXEC PGM=PAGENT...
//   PARM='.../-c /etc/pagent.pok.conf'
...
//STDENV   DD PATH='/etc/pagent.env'
...
```

```
TcpImage TCPIPA /etc/pagent.poka.conf
TcpImage TCPIPB /etc/pagent.pokb.conf
TcpImage TCPIPC /etc/pagent.pokc.conf
```

```
ReadFromDirectory
{
  LDAP_Server 9.12.6.63
  LDAP_ProtocolVersion 3
  LDAP_SchemaVersion 3
  SearchPolicyBaseDN pg=TCPIPA, g=policy..
}
```

Figure 4-3 Policy agent configuration - Download policies from the LDAP Server

In this case PAGENT takes the role of an LDAP client. In Figure 4-3 you see that the address of the LDAP Server to be referenced is identified in the configuration file read by each stack. For TCPIPA, you can see (below) the PAGENT configuration file, /etc/pagent.poka.conf, that will be read by TCPIPA.

```
#   SecureWay Communications Server IP for OS/390
#   5647-A01 (C) Copyright IBM Corp. 1998, 1999.
#   Licensed Materials - Property of IBM
#
#        /etc/pagent.poka.conf  (PAGENT policy configuration)
ReadFromDirectory                                              1
{
     LDAP_Server                  9.12.6.63                    2
     LDAP_Port                    3389                         3
     LDAP_DistinguishedName       cn=LDAP
     LDAP_Password                secret
     LDAP_ProtocolVersion         3                            4
     LDAP_SchemaVersion           3                            5
     SearchPolicyBaseDN           o=ITSO,c=US                  6
     SearchPolicyKeyword          "rule1"                      7
#    LDAP_Backupserver                                         8
     LDAP_SessionPersistent       Yes                          9
}


SetSubnetPrioTosMask
{
     SubnetAddr 0.0.0.0
     SubnetTosMask 11100000
     PriorityTosMapping 1 11100000 0
     PriorityTosMapping 1 11000000 0
     PriorityTosMapping 1 10100000 0
     PriorityTosMapping 1 10000000 0
     PriorityTosMapping 2 01100000 0
     PriorityTosMapping 3 01000000 0
     PriorityTosMapping 4 00100000 0
     PriorityTosMapping 4 00000000 0
}
```

The ReadFromDirectory statement initializes the Policy Agent as an LDAP client. Using the ReadFromDirectory statement 1, we can specify the LDAP Server's host name or IP address 2, the LDAP port to connect to 3, the LDAP protocol version used 4, the policy version schema 5, and the distinguished name of the subtree in the directory containing the policies 6.

Using the SearchPolicyRuleKeyWord 7, you can specify a string used to select a subset of the policies under the specified base tree 6. You can define a backup LDAP Server 8 if the primary LDAP Server does not respond; the default is to not use a backup LDAP Server.

You can choose if the LDAP session with the directory server should be kept open or closed during an update interval time by using the LDAP_SessionPersistent 9 parameter. If the LDAP session update interval is small, specify yes for this parameter to reduce the overhead of opening the session for each query.

4.3.1 Expressing the policies to LDAP

The following pages are the same policies as those shown in the configuration file in 4.2, "Coding policy definitions in a configuration file" on page 77, only here they are shown in the format necessary for them to be stored in a LDAP Server.

```
dn:ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:organizationalUnit
ou:zqosMgrPolicies

dn:cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:policyRepository
ibm-policyRepositoryName:policyRepository

dn:cn=actionRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:actionRepository
ibm-policyRepositoryName:actionRepository

dn:cn=serverRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:serverRepository
ibm-policyRepositoryName:serverRepository

dn:cn=sysplexServerRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:sysplexServerRepository
ibm-policyRepositoryName:sysplexServerRepository

dn:cn=clientRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:clientRepository
```

```
ibm-policyRepositoryName:clientRepository

dn:cn=sysplexClientRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:sysplexClientRepository
ibm-policyRepositoryName:sysplexClientRepository

dn:cn=userRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:userRepository
ibm-policyRepositoryName:userRepository

dn:cn=groupRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:groupRepository
ibm-policyRepositoryName:groupRepository

dn:cn=applRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:applRepository
ibm-policyRepositoryName:applRepository

dn:cn=timeRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:timeRepository
ibm-policyRepositoryName:timeRepository

dn:cn=protoRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:protoRepository
ibm-policyRepositoryName:protoRepository

dn:cn=outAddrRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:outAddrRepository
ibm-policyRepositoryName:outAddrRepository

dn:cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyGroup
cn:policyRules
ibm-policyGroupName:policyRules

dn:cn=sysplexPolicyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyGroup
objectclass:ibm-policyGroupLoadDistributionAuxClass
cn:sysplexPolicyRules
ibm-policyGroupName:sysplexPolicyRules
ibm-policyGroupForLoadDistribution:TRUE

dn:cn=batch1, cn=actionRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
```

```
objectclass:ibm-policyInstance
objectclass:ibm-policyActionInstance
objectclass:ibm-policyActionAuxClass
objectclass:ibm-serviceCategoriesAuxClass
cn:batch1
ibm-policyActionName:batch1
ibm-policyScope:DataTraffic
ibm-OutgoingTOS:01000000
ibm-MaxRate:0
ibm-DiffServInProfileRate:0
ibm-DiffServInProfileTokenBucket:0
ibm-DiffServExcessTrafficTreatment:BestEffort
ibm-DiffServOutProfileTransmittedTOSByte:00000000
ibm-MinRate:0
ibm-MaxDelay:0
ibm-MaxConnections:0
ibm-Permission:Allowed

dn:cn=interactive1, cn=actionRepository, cn=policyRepository, ou=zqosMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyActionInstance
objectclass:ibm-policyActionAuxClass
objectclass:ibm-serviceCategoriesAuxClass
cn:interactive1
ibm-policyActionName:interactive1
ibm-policyScope:DataTraffic
ibm-OutgoingTOS:10000000
ibm-MaxRate:0
ibm-DiffServInProfileRate:0
ibm-DiffServInProfileTokenBucket:0
ibm-DiffServExcessTrafficTreatment:BestEffort
ibm-DiffServOutProfileTransmittedTOSByte:00000000
ibm-MinRate:0
ibm-MaxDelay:0
ibm-MaxConnections:0
ibm-Permission:Allowed

dn:cn=networkcontrol, cn=actionRepository, cn=policyRepository, ou=zqosMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyActionInstance
objectclass:ibm-policyActionAuxClass
objectclass:ibm-serviceCategoriesAuxClass
cn:networkcontrol
ibm-policyActionName:networkcontrol
ibm-policyScope:DataTraffic
ibm-OutgoingTOS:11100000
ibm-MaxRate:0
ibm-DiffServInProfileRate:0
ibm-DiffServInProfileTokenBucket:0
ibm-DiffServExcessTrafficTreatment:BestEffort
ibm-DiffServOutProfileTransmittedTOSByte:00000000
ibm-MinRate:0
ibm-MaxDelay:0
ibm-MaxConnections:0
ibm-Permission:Allowed
```

```
dn:cn=Tscsrv, cn=actionRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyActionInstance
objectclass:ibm-policyActionAuxClass
objectclass:ibm-serviceCategoriesAuxClass
cn:Tscsrv
ibm-policyActionName:Tscsrv
ibm-policyScope:DataTraffic
ibm-OutgoingTOS:01100000
ibm-MaxRate:120
ibm-DiffServInProfileRate:0
ibm-DiffServInProfileTokenBucket:0
ibm-DiffServExcessTrafficTreatment:BestEffort
ibm-DiffServOutProfileTransmittedTOSByte:00000000
ibm-MinRate:0
ibm-MaxDelay:10
ibm-MaxConnections:0
ibm-Permission:Allowed

dn:cn=ftp, cn=serverRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-networkingPolicyConditionAuxClass
objectclass:ibm-hostConditionAuxClass
objectclass:ibm-applicationConditionAuxClass
cn:ftp
ibm-policyConditionName:ftp
ibm-sourcePortRange:20:21

dn:cn=telnet, cn=serverRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-networkingPolicyConditionAuxClass
objectclass:ibm-hostConditionAuxClass
objectclass:ibm-applicationConditionAuxClass
cn:telnet
ibm-policyConditionName:telnet
ibm-sourcePortRange:23:23

dn:cn=ee1, cn=serverRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-networkingPolicyConditionAuxClass
objectclass:ibm-hostConditionAuxClass
objectclass:ibm-applicationConditionAuxClass
cn:ee1
ibm-policyConditionName:ee1
ibm-sourceIPAddressRange:3-172.16.232.1-172.16.232.1
ibm-sourcePortRange:12000:12000

dn:cn=ee2, cn=serverRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
```

```
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-networkingPolicyConditionAuxClass
objectclass:ibm-hostConditionAuxClass
objectclass:ibm-applicationConditionAuxClass
cn:ee2
ibm-policyConditionName:ee2
ibm-sourceIPAddressRange:3-172.16.252.254-172.16.252.254
ibm-sourcePortRange:12000:12000

dn:cn=ftp, cn=clientRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-networkingPolicyConditionAuxClass
objectclass:ibm-hostConditionAuxClass
objectclass:ibm-applicationConditionAuxClass
cn:ftp
ibm-policyConditionName:ftp
ibm-destinationPortRange:20:21

dn:cn=ee1, cn=clientRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-networkingPolicyConditionAuxClass
objectclass:ibm-hostConditionAuxClass
objectclass:ibm-applicationConditionAuxClass
cn:ee1
ibm-policyConditionName:ee1
ibm-destinationIPAddressRange:3-172.16.232.1-172.16.232.1
ibm-destinationPortRange:12000:12000

dn:cn=ee2, cn=clientRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-networkingPolicyConditionAuxClass
objectclass:ibm-hostConditionAuxClass
objectclass:ibm-applicationConditionAuxClass
cn:ee2
ibm-policyConditionName:ee2
ibm-destinationIPAddressRange:3-172.16.252.254-172.16.252.254
ibm-destinationPortRange:12000:12000

dn:cn=telnet, cn=clientRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-networkingPolicyConditionAuxClass
objectclass:ibm-hostConditionAuxClass
objectclass:ibm-applicationConditionAuxClass
cn:telnet
ibm-policyConditionName:telnet
ibm-destinationPortRange:23:23
```

```
dn:cn=protocol-1, cn=protoRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-networkingPolicyConditionAuxClass
objectclass:ibm-applicationConditionAuxClass
cn:protocol-1
ibm-policyConditionName:protocol-1
ibm-protocolNumberRange:7:7

dn:cn=protocol-2, cn=protoRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-networkingPolicyConditionAuxClass
objectclass:ibm-applicationConditionAuxClass
cn:protocol-2
ibm-policyConditionName:protocol-2
ibm-protocolNumberRange:17:17

dn:cn=protocol-3, cn=protoRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-networkingPolicyConditionAuxClass
objectclass:ibm-applicationConditionAuxClass
cn:protocol-3
ibm-policyConditionName:protocol-3
ibm-protocolNumberRange:89:89

dn:cn=ftpd, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRule
objectclass:ibm-policySubtreesPtrAuxClass
cn:ftpd
ibm-policyRuleName:ftpd
ibm-policyRuleEnabled:1
ibm-policyRuleConditionListType:2
ibm-policyRulePriority:255

dn:cn=actAssoc-0, cn=ftpd, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleActionAssociation
objectclass:ibm-policyActionAuxClass
cn:actAssoc-0
ibm-policyActionName:actAssoc-0
ibm-policyActionOrder:0
ibm-policyActionDN:cn=batch1, cn=actionRepository, cn=policyRepository, ou=zqosMgrPolicies,
o=ITSO,c=US

dn:cn=servAssoc-0-0, cn=ftpd, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:servAssoc-0-0
ibm-policyConditionName:servAssoc-0-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:1
```

```
ibm-policyConditionDN:cn=ftp, cn=serverRepository, cn=policyRepository, ou=zqosMgrPolicies,
o=ITSO,c=US

dn:cn=cliAssoc-0-0, cn=ftpd, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:cliAssoc-0-0
ibm-policyConditionName:cliAssoc-0-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:2
ibm-policyConditionDN:cn=ftp, cn=clientRepository, cn=policyRepository, ou=zqosMgrPolicies,
o=ITSO,c=US

dn:cn=protoAssoc-0-0, cn=ftpd, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:protoAssoc-0-0
ibm-policyConditionName:protoAssoc-0-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:6
ibm-policyConditionDN:cn=protocol-1, cn=protoRepository, cn=policyRepository,
ou=zqosMgrPolicies, o=ITSO,c=US

dn:cn=telnet, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRule
objectclass:ibm-policySubtreesPtrAuxClass
cn:telnet
ibm-policyRuleName:telnet
ibm-policyRuleEnabled:1
ibm-policyRuleConditionListType:2
ibm-policyRulePriority:254

dn:cn=actAssoc-1, cn=telnet, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleActionAssociation
objectclass:ibm-policyActionAuxClass
cn:actAssoc-1
ibm-policyActionName:actAssoc-1
ibm-policyActionOrder:0
ibm-policyActionDN:cn=interactive1, cn=actionRepository, cn=policyRepository,
ou=zqosMgrPolicies, o=ITSO,c=US

dn:cn=servAssoc-1-0, cn=telnet, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:servAssoc-1-0
ibm-policyConditionName:servAssoc-1-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:1
ibm-policyConditionDN:cn=telnet, cn=serverRepository, cn=policyRepository,
ou=zqosMgrPolicies, o=ITSO,c=US

dn:cn=cliAssoc-1-0, cn=telnet, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:cliAssoc-1-0
ibm-policyConditionName:cliAssoc-1-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:2
```

```
ibm-policyConditionDN:cn=telnet, cn=clientRepository, cn=policyRepository,
ou=zqosMgrPolicies, o=ITSO,c=US

dn:cn=protoAssoc-1-0, cn=telnet, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:protoAssoc-1-0
ibm-policyConditionName:protoAssoc-1-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:6
ibm-policyConditionDN:cn=protocol-1, cn=protoRepository, cn=policyRepository,
ou=zqosMgrPolicies, o=ITSO,c=US

dn:cn=ee, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRule
objectclass:ibm-policySubtreesPtrAuxClass
cn:ee
ibm-policyRuleName:ee
ibm-policyRuleEnabled:1
ibm-policyRuleConditionListType:2
ibm-policyRulePriority:253

dn:cn=actAssoc-2, cn=ee, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleActionAssociation
objectclass:ibm-policyActionAuxClass
cn:actAssoc-2
ibm-policyActionName:actAssoc-2
ibm-policyActionOrder:0
ibm-policyActionDN:cn=Tscsrv, cn=actionRepository, cn=policyRepository, ou=zqosMgrPolicies,
o=ITSO,c=US

dn:cn=servAssoc-2-0, cn=ee, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:servAssoc-2-0
ibm-policyConditionName:servAssoc-2-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:1
ibm-policyConditionDN:cn=ee1, cn=serverRepository, cn=policyRepository, ou=zqosMgrPolicies,
o=ITSO,c=US

dn:cn=servAssoc-2-1, cn=ee, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:servAssoc-2-1
ibm-policyConditionName:servAssoc-2-1
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:1
ibm-policyConditionDN:cn=ee2, cn=serverRepository, cn=policyRepository, ou=zqosMgrPolicies,
o=ITSO,c=US

dn:cn=cliAssoc-2-0, cn=ee, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:cliAssoc-2-0
ibm-policyConditionName:cliAssoc-2-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:2
```

```
ibm-policyConditionDN:cn=ee1, cn=clientRepository, cn=policyRepository, ou=zqosMgrPolicies,
o=ITSO,c=US

dn:cn=cliAssoc-2-1, cn=ee, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:cliAssoc-2-1
ibm-policyConditionName:cliAssoc-2-1
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:2
ibm-policyConditionDN:cn=ee2, cn=clientRepository, cn=policyRepository, ou=zqosMgrPolicies,
o=ITSO,c=US

dn:cn=protoAssoc-2-0, cn=ee, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:protoAssoc-2-0
ibm-policyConditionName:protoAssoc-2-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:6
ibm-policyConditionDN:cn=protocol-2, cn=protoRepository, cn=policyRepository,
ou=zqosMgrPolicies, o=ITSO,c=US

dn:cn=ospf, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRule
objectclass:ibm-policySubtreesPtrAuxClass
cn:ospf
ibm-policyRuleName:ospf
ibm-policyRuleEnabled:1
ibm-policyRuleConditionListType:2
ibm-policyRulePriority:252

dn:cn=actAssoc-3, cn=ospf, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleActionAssociation
objectclass:ibm-policyActionAuxClass
cn:actAssoc-3
ibm-policyActionName:actAssoc-3
ibm-policyActionOrder:0
ibm-policyActionDN:cn=networkcontrol, cn=actionRepository, cn=policyRepository,
ou=zqosMgrPolicies, o=ITSO,c=US

dn:cn=protoAssoc-3-0, cn=ospf, cn=policyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:protoAssoc-3-0
ibm-policyConditionName:protoAssoc-3-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:6
ibm-policyConditionDN:cn=protocol-3, cn=protoRepository, cn=policyRepository,
ou=zqosMgrPolicies, o=ITSO,c=US
```

4.3.2 Considerations when opening SSL connections

In order for the LDAP client to communicate with an LDAP Server over an SSL-protected TCP/IP socket connection, the LDAP Server must transmit a certificate to the LDAP client and, optionally, the client can transmit its certificate to the LDAP Server. The LDAP client and server must verify that the certificates they receive are valid. Once the LDAP client and server have determined the validity of the certificates provided to them, SSL-protected communications are established between them.

You can create the certificates for your organization using self-signed certificates, and in this situation you are the CA authority. Or you can obtain certificates signed by an external CA authority, such as VeriSign.

When you create your self-signed certificate or you receive the certificate from a CA authority, you must store the certificate in a keyring database. This should be a RACF database or an HFS file created with the GSKKYMAN utility.

We defined the LDAP Server to allow the LDAP client to validate the LDAP Server and the LDAP Server to validate the LDAP client if the client sends its digital certificate on the initial contact between the client and the server. And we used HFS keyring databases to store the self-signed LDAP Server's certificate and the PAGENT certificate.

To accomplish this, follow the steps below:

1. Define a keyring database for the LDAP Server, as an HFS file using the GSKKYMAN utility.
2. Add a self-signed certificate for the LDAP Server in the keyring database.
3. In the LDAP configuration file, add the following statements for the LDAP Server to support the SSL communications.

```
port 389
secureport 636                              1
# security <sslonly|ssl|none>
security ssl                                2
# sslAuth <serverClientAuth|serverAuth>
sslAuth serverClientAuth                    3
sslKeyRingFile /etc/ldap/ldapring.kdb       4
sslKeyRingPWStashFile /etc/ldap/ldapring.sth 5
sslKeyRingFilePW r2615                      6
```

   We used the TCP port 636 1 for secure connections, enabled SSL 2, and defined the security method 3, which indicates both client and server authentication are supported. In addition to those parameters above, the keyring database 4, stash file names 5, and the password 6 to access to the database have to be defined.

4. Define a keyring database used by the Policy Agent. We created an HFS file using the GSKKYMAN utility.
5. In the PAGENT configuration file, configure the following statements so that PAGENT uses SSL to communicate with LDAP.

```
# Define the TCP port number for the SSL connections
  LDAP_Port 636                             7
# Definitions for the LDAP SSL communications
 LDAP_SSL                                   8
 {
    LDAP_SSLKeyringFile /etc/ldap/pagent.kdb 9
    LDAP_SSLKeyringPassword r2615           10
#   LDAP_SSLName PAGENT_RA28                11
 }
```

The port parameter (7) is not directly related to an SSL connection, but we had to specify the port number used to open the SSL connections to the well-known port 398. The LDAP_SSL (8) statement enables the SSL communication between PAGENT and the LDAP Server, and has to be enclosed by `{ }`. The following parameters related to the SSL communication are supported:

- LDAP_SSLKeyringFile 9, which defines the keyring file that contains the certificates of the Certificate Authorities that are trusted by the client. When client authentication is required, you must have a public key and the associated certificate stored in the key database.
- LDAP_SSLKeyringPassword 10, which specifies the password to access to the keyring file.
- LDAP_SSLName, which specifies the label assigned when creating your private key/certificate pair with **gskkyman** or **racdcert**. You can read more on this by referencing *Communications Server for z/OS V1R2 TCP/IP Implementation Guide, Volume 7: Security,* SG24-6840. When the client is authenticated by the LDAP Server, this parameter is required. The value is case sensitive.

6. Export the LDAP Server self-signed certificate from the LDAP key database into an HFS file.
7. Store the LDAP self-signed certificate into the PAGENT keyring database.

When client authentication is required, do the following actions additionally. Note that we used a self-signed certificate also for PAGENT.

1. Define a self-signed certificate in the PAGENT keyring database.
2. Export the PAGENT self-signed certificate from the PAGENT keyring database, and save it in an HFS file.
3. Import the PAGENT self-signed certificate into the LDAP keyring database.
4. Configure the LDAP_SSLName 11 parameter in the PAGENT configuration file to let the Policy Agent send its certificate. This is the label name you define for your self-signed certificate.

Note: The LDAP_SSLName value must be defined without blanks inside.

The SSL support is now provided by the System Secure Sockets Layer (System SSL) element of z/OS and it includes a database utility called GSKKYMAN used to create the keyring file; see *z/OS V1R2.0 System Secure Socket Layer Programming,* SC24-5901.

For more information about the LDAP Server configuration for the SSL communication, consult *z/OS V1R2.0 Security Server LDAP Server Administration and Use,* SC24-5923.

4.3.3 Policy agent log file

When you start the Policy Agent as a started task, the output message written to stdout and stderr goes to the data set or file specified with SYSPRINT or SYSOUT DD, respectively. But normally, PAGENT does not write output to stdout or stderr. Instead, output is written to the log file, which can be specified by the PAGENT_LOG_FILE environment variable and defaults to /tmp/pagent.log. When the -d parameter is specified, however, output is also written to stdout. The log file is created when the policy agent is activated, if it does not exist.

You can check whether your policies have been installed in the associated stacks as defined. In this scenario, some policies have been configured in the image configuration file (the second level configuration file), and some have been stored in the LDAP Server. See “The image configuration file (the policy and LDAP definitions)” on page 266, for the image configuration file user ID in this scenario.

The following figures show you the messages when the policies are being installed. You can specify option -d at startup, so that additional messages will be displayed.

```
INFO  :.log_init[31]: Log File Size = 300, Number of Log Files = 3
INFO  :.main[31]: OS/390 Service Policy Agent
LOG   :.main[31]: EZZ8431I PAGENT STARTING
INFO  :.main[31]: Use environment PAGENT_CONFIG_FILE = /etc/pagent.r2615.conf 1
INFO  :.main[31]: Using log level 511    2

INFO  :..reg_process[30]: registering process with the system
INFO  :..reg_process[30]: attempt OS/390 registration
INFO  :..reg_process[30]: return from registration rc=0
INFO  :....mailslot_create[30]: creating mailslot for timer
INFO  :...mailbox_register[30]: mailbox allocated for timer
INFO  :...mailslot_create[30]: creating mailslot for terminate
INFO  :..mailbox_register[30]: mailbox allocated for terminate
INFO  :...mailslot_create[30]: creating mailslot for (broken) pipe
INFO  :..mailbox_register[30]: mailbox allocated for pipe
INFO  :...mailslot_create[30]: creating mailslot for process abend
INFO  :..mailbox_register[30]: mailbox allocated for abend
INFO  :...mailslot_create[30]: creating mailslot for process abort
INFO  :..mailbox_register[30]: mailbox allocated for abort
INFO  :...mailslot_create[30]: creating mailslot for reconfig
INFO  :..mailbox_register[30]: mailbox allocated for config
INFO  :...mailslot_create[30]: creating mailslot for process quit
INFO  :..mailbox_register[30]: mailbox allocated for quit
INFO  :..pagent_mvs_command_handler[29]: Command handler thread active
INFO  :..profile_initialize[30]: Pagent threads created installation(/flush)
                                      and initialization.
INFO  :..check_main_config_file[28]: Main config file thread active
INFO  :...pinit_create_tmpFile[28]: Creating temporary working file
                                     /tmp/TCPIPC.Pagent.tmp
INFO  :...mailslot_create[30]: creating mailslot for dump
INFO  :..mailbox_register[30]: mailbox allocated for dump
INFO  :..listen_thread[27]: PAPI server thread active
INFO  :..pinit_init_tcpimages[28]: processed TcpImage statement:
                 TcpImage TCPIPC /etc/pagent.r2615c.ldap.conf  FLUSH 600   3

LOG   :.main[30]: EZZ8432I PAGENT INITIALIZATION COMPLETE
```

Figure 4-4 PAGENT startup log

During the initialization phase, PAGENT reads the policy agent configuration file and extracts the necessary information. In Figure 4-4, the HFS file /etc/pagent.r2615.conf 1 has been specified in the startup option for PAGENT.

The log level has been set to 511 2 so that you will see additional information.

3 Start processing the TcpImage statement.

```
INFO  :..pinit_init_tcpimages[28]: Main config file refresh interval = 600 seconds
INFO  :.check_main_config_file[28]: Start up a config event update thread
INFO  :.check_main_config_file[28]: Finish starting policy profile
                       installation(/flush) and initialization.
INFO  :.check_config_files[26]: Config processing thread active for image
                               'TCPIPC', index 0
INFO  :.config_files_update_event[25]: Config files update event thread active
INFO  :...settcpimage[26]: Associate with TCP/IP image name = TCPIPC   4
INFO  :.config_files_update_event[25]: IPC message queue id obtained, qid =30
INFO  :..FlushAllPolicies[26]: Start first by flushing all policies    5
INFO  :...profile_delete_ALL_ServiceClass{26]: Image name: TCPIPC      4
INFO  :...profile_delete_ALL_ServiceClass[26]: Finished deleting ALL
                                             Service Class in image 0  6
INFO  :...profile_delete_ALL_PolicyRule[26]: Image name:  TCPIPC
INFO  :...profile_delete_ALL_PolicyRule[26]: Finished deleting ALL
                                                         Policy Rules  7
INFO  :..pinit_fetch_policy_profile[26]: Processing policy config data file:
                        /etc/pagent.r2615c.ldap.conf for image TCPIPC  8
INFO  :...processing_Stmt_UseLDAPRules[26]: (Re)starting LDAP processing
                                              thread for image 'TCPIPC'
INFO  :...processing_Stmt_ServicePolicyRu[26]: processing:  policyRule telnetRule 9

INFO  :.ReadLdapRules[24]: LDAP processing thread active for image 'TCPIPC', index 0
INFO  :...processing_Stmt_ServiceCategori[26]: processing:  policyAction telnetAction 10

INFO  :...UpdateSCProfileData[26]: Updating SC profile for caller id: 1
INFO  :....computePolicyRuleWeight[26]: Weight computed for PR telnetRule is: 2
INFO  :...UpdatePRProfileData[26]: Updating PR profile for caller id: 1
INFO  :....computePolicyRuleWeight[26]: Weight computed for PR telnetRule is: 2
INFO  :...profile_install_A_ServiceClass[26]: Service Class: telnetAction
INFO  :...profile_install_A_ServiceClass[26]: Finished installing Service Class:
                                                              telnetAction
TRACE :....process_time_condition[26]: PR=telnetRule active, next check in 455 minutes
INFO  :...profile_install_A_PolicyRule[26]: Finished installing policy rule: telnetRule
INFO  :..pinit_fetch_policy_profile[26]: Finish processing above policy config file 11
```

Figure 4-5 PAGENT configuration file processing

Since we have the FLUSH parameter configured, PAGENT will try to delete all existing policies 5 and 6 and rules 7 from the associated TCP/IP stack, which is TCPIPC 4 in this sample. After deleting all policies and rules 7, PAGENT will start to process the PAGENT configuration file 8 and start to process the PolicyRule 9 and PolicyAction 10 defined in the PAGENT configuration file; 11 the process of configuration file finishes.

Figure 4-6 on page 96 shows the process of initializing the policy agent as an LDAP client. PAGENT is searching for all policies stored in the LDAP Server 1, but it only looks for the objects associated with RA2615 as defined in the /etc/pagent.r2615c.ldap configuration file; see 4.3, “PAGENT configuration file as an LDAP client” on page 81.

We used Policy Schema 2 version 2. PAGENT started to process the PolicyRule 3 and the PolicyAction 4. For every policy rule a weight is defined 5, which is the PolicyRulePriority defined in the policyRule statement. If you do not define this statement, PAGENT will define a value between 1 and 100. If you define ibm-policyRulePriority value for a policy in the LDAP Server, PAGENT will take this value and add the value of 100 6 to create a weight value. First PAGENT processes all the PolicyActions, then installs them into TCPIPC. The policy agent installed the service class ftpaction into the stack TCPIPC 7.

```
INFO :..settcpimage[24]: Associate with TCP/IP image name = TCPIPC
INFO :.ReadLdapRules[24]: Contacting LDAP server 172.16.250.3 on port 389        1
INFO :.ReadLdapRules[24]: Processing version 2 schema                             2
INFO :...search_ldap[24]: Searching for policies with base pg=RA2615, g=policy,
                      o=IBM_US, c=US, filter (&(objectClass=*)), scope subtree
INFO :..sla_ldap_get_v2_policies[24]: 1 objects returned from LDAP search
TRACE :...process_ldap_group[24]: Search group contains object pg=RA2615, g=policy,
                      o=IBM_US, c=US
INFO :..sla_ldap_get_v2_policies[24]: Processing policy group pg=TCPIP,g=policy,
                     o=IBM_US,c=US
INFO :...search_ldap[24]: Searching for policies with base pg=TCPIP,g=policy,
                     o=IBM_US,c=US, filter (&(objectClass=*)), scope base
INFO :..sla_ldap_get_v2_policies[24]: Processing policy group pg=TCPIPC,g=policy,
                     o=IBM_US,c=US
INFO :...search_ldap[24]: Searching for policies with base pg=TCPIPC,g=policy,
                     o=IBM_US,c=US, filter (&(objectClass=*)), scope base
.....................................................................................
INFO :..sla_ldap_get_v2_policies[24]: Processing policy rule pr=ftprule,pg=TCPIPC, 3
                    g=policy,o=IBM_US,c=US
INFO :...search_ldap[24]: Searching for policies with base pr=ftprule,pg=TCPIPC,
                    g=policy,o=IBM_US,c=US, filter (&(objectClass=*)), scope base
INFO :....search_ldap[24]: Searching for policies with base pc=cond1,pg=TCPIPC,
                    g=policy,o=IBM_US,c=US, filter (&(objectClass=*)), scope base
INFO :....search_ldap[24]: Searching for policies with base pa=ftpaction,pg=TCPIPC,
                    g=policy,o=IBM_US,c=US, filter (&(objectClass=*)), scope base
......................................................................................
INFO :..sla_ldap_get_v2_policies[24]: Processing policy action pa=ftpaction,pg=TCPIPC, 4
                   g=policy,o=IBM_US,c=US
INFO :..UpdateSCProfileData[24]: Updating SC profile for caller id: 2
INFO :...computePolicyRuleWeight[24]: Weight computed for PR ospf is: 2          5
INFO :...computePolicyRuleWeight[24]: Weight computed for PR ftpd is: 3
INFO :...computePolicyRuleWeight[24]: Weight computed for PR telnetd is: 3
INFO   :...computePolicyRuleWeight[24]: Weight computed for PR ftprule is: 102 6
INFO   :..UpdatePRProfileData[24]: Updating PR profile for caller id: 2
INFO   :...computePolicyRuleWeight[24]: Weight computed for PR ftprule is: 102
INFO   :...computePolicyRuleWeight[24]: Weight computed for PR ftpd is: 3
INFO   :...computePolicyRuleWeight[24]: Weight computed for PR telnetd is: 3
INFO   :...computePolicyRuleWeight[24]: Weight computed for PR ospf is: 2
INFO   :...computePolicyRuleWeight[24]: Weight computed for PR telnetRule is: 2
.....................................................................................
INFO   :....settcpimage[24]: Associate with TCP/IP image name = TCPIPC
INFO   :...profile_install_A_ServiceClass[24]: Service Class: ftpaction
INFO   :...profile_install_A_ServiceClass[24]: Finished installing Service
                                                                  Class: ftpaction   7
```

Figure 4-6 PAGENT configuration file processing as an LDAP client

```
TRACE :....process_time_condition[24]: PR=ftprule active, next check in 214 minutes
INFO  :....settcpimage[24]: Associate with TCP/IP image name = TCPIPC     1
INFO  :...profile_install_A_PolicyRule[24]: Finished installing policy rule: ftprule 2
TRACE :....process_time_condition[24]: PR=ftpd active, next check in 214 minutes
INFO  :....settcpimage[24]: Associate with TCP/IP image name = TCPIPC
INFO  :...profile_install_A_PolicyRule[24]: Finished installing policy rule: ftpd 2
TRACE :....process_time_condition[24]: PR=telnetd active, next check in 214 minutes
INFO  :....settcpimage[24]: Associate with TCP/IP image name = TCPIPC
INFO  :...profile_install_A_PolicyRule[24]: Finished installing policy rule: telnetd 2
TRACE :....process_time_condition[24]: PR=ospf active, next check in 214 minutes
INFO  :....settcpimage[24]: Associate with TCP/IP image name = TCPIPC
INFO  :...profile_install_A_PolicyRule[24]: Finished installing policy rule: ospf 2
INFO  :...search_ldap[24]: Searching for policies with base pg=RA2615, g=policy,
        o=IBM_US, c=US, filter (&(objectClass=setsubnetpriotosmask)), scope subtree
INFO  :.process_rqsts[23]: PAPI client thread active for connection 3
INFO  :..paapi_search[23]: Found Match to Policy Entry Name = ftprule
INFO  :..paapi_search[23]: Found Match to Policy Entry Name = ftpaction
INFO  :..paapi_search[23]: Found Match to Policy Entry Name = ftpd
INFO  :..paapi_search[23]: Found Match to Policy Entry Name = batch1
INFO  :..paapi_search[23]: Found Match to Policy Entry Name = telnetd
INFO  :..paapi_search[23]: Found Match to Policy Entry Name = interactive1
INFO  :..paapi_search[23]: Found Match to Policy Entry Name = telnetRule
INFO  :..paapi_search[23]: Found Match to Policy Entry Name = telnetAction
INFO  :..paapi_search[23]: Found Match to Policy Entry Name = ospf
INFO  :..paapi_search[23]: Found Match to Policy Entry Name = networkcontrol
```

Figure 4-7 Policy rules installed into TCPIPC

Figure 4-7 shows that PAGENT is processing the Service Policy rules. PAGENT sets up the time at which the policy rules have to be applied. PAGENT reads through all the policy rule entries first, then installs them into TCPIPC 1. PAGENT installed policy rules ftprule, ftpd, telnetd, and ospf 2 into the stack TCPIPC.

```
TRACE :....process_time_condition[24]: PR=ftprule inactive, next check in 840 minutes  3
INFO  :....settcpimage[24]: Associate with TCP/IP image name = TCPIPC                  4
INFO  :...profile_delete_A_PolicyRule[24]: Policy Rule to be deleted:  ftprule
INFO  :...profile_delete_A_PolicyRule[24]: Policy Rule was deleted by name
INFO  :...profile_delete_A_PolicyRule[24]: Finished deleting policy rule: ftprule     5

TRACE  :......process_time_condition[30]: PR=ftprule active, next check in  840 minutes
TRACE  :.....PolicyRuleTimer_expire[30]: PolicyRule ftprule changes from Active state 0,
                                                                         to state 1  6
INFO   :.......settcpimage[30]: Associate with TCP/IP image name = TCPIPC
INFO   :......profile_install_A_PolicyRule[30]: Finished installing policy rule: ftprule
```

Figure 4-8 Time expired and deleting rule

Figure 4-8 shows that after the policy rule timer expired 3, PAGENT deleted the policy rule ftprule 5 associated with the stack TCPIPC 4. On the other hand, when the time period restarts, PAGENT changes the status from inactive 0 to active 1 6 and the policy rule ftprule will be enabled 7.

Part 3

Implementing Quality of Service

5

Understanding Quality of Service

As application data, voice and video traffic converge over a single utilitarian network based on the Internet Protocol, it becomes increasingly important to provide the exact services that each traffic type requires. A network that provides services in terms of the latency, jitter, and packet delivery required by each application and based on proper, established business policies will meet the needs of the organization more efficiently. The result is a cost-effective, converged network offering increased productivity and competitive advantage by allowing applications to be developed, enhanced, and deployed faster than ever.

This chapter will help you to define Quality of Service (QoS) levels within a Cisco network and map z/OS systems and application traffic to the services based on their requirements. The chapter is organized according to the following topics:

- Overview of QoS protocols
- Steps in QoS deployment
- QoS on the z/OS Communications Server
- Ensuring QoS across the Cisco network
- Managing QoS

Included is information to help you configure the QoS features of Communications Server for z/OS to operate with the QoS features of the Cisco network.

5.1 Overview of QoS protocols

Various protocols have been developed to tackle the problem of providing end-to-end services based on application requirements. SNA and APPN have evolved over time with the idea of class of service according to application requirements central to its development.

Today's networks are IP based. Initially, IP was defined to provide only a best-effort delivery service. A network built using no additional QoS mechanisms is still very robust and services many disparate applications, as proven by the popularity and scale of the global Internet. However, as enterprises make heavier use of the Internet and build their own intranets, bandwidth will inevitably become constrained. No longer will the network be able to offer the desired services to each and every application. Convergence beyond data adds new requirements for traffic delivery. Voice, video, and other digitally encoded streams have real-time transport service requirements that cannot be accommodated without supplemental protocols and policies.

QoS protocols, then, are necessary to allow intelligent packet forwarding decisions to be made based on the end-to-end service goals of the application.

5.1.1 Service models

Remember, service levels apply to applications and traffic streams on an end-to-end basis. This is an important point to remember when designing a network with QoS. It is also fundamental to the ability of a network, or a subnetwork, to provide different levels of service without introducing complexity that cannot be managed.

Let us start with a basic definition of three commonly referred to models of end-to-end service (Figure 5-1 on page 103):

- Best-effort service

  Best-effort service is the type of service provided by all general IP-based networks. The network will deliver data, on a first-in, first-out basis as long as resources are available to do so. No guarantees or assurances are made with respect to delay, packet loss, or throughput. You could say a best-effort service lacks any QoS mechanisms.

- Differentiated Services

  Differentiated Services involves the handling of individual packets or flows within a network node. Each packet is associated with a particular class of service. Each node along the network path handles packets in a cooperative manner according to a common set of rules resulting in end-to-end service classes.

- Integrated or Reserved Services

  Integrated Services, also known as *Reserved or Guaranteed Services*, provides the bandwidth and delay characteristics as requested by the application or configured for specific types of traffic.

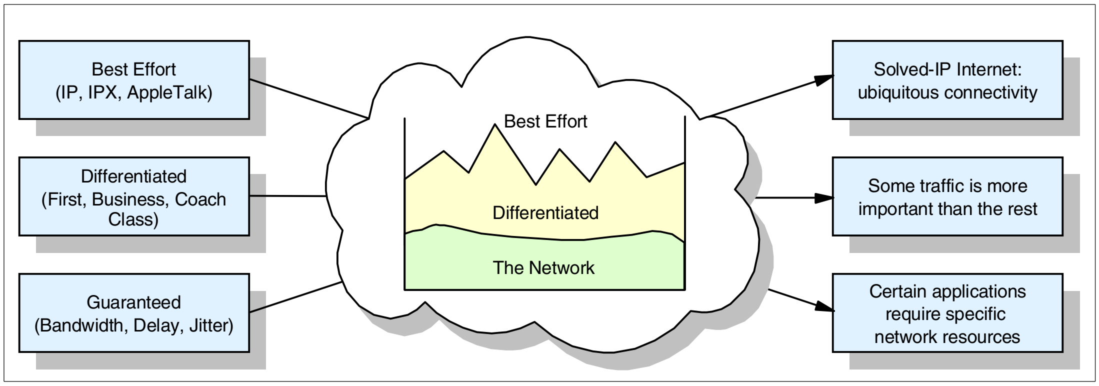

Best Effort
(IP, IPX, AppleTalk)
Differentiated
(First, Business, Coach Class)
Guaranteed
(Bandwidth, Delay, Jitter)
Best Effort
Differentiated
The Network
Solved-IP Internet: ubiquitous connectivity
Some traffic is more important than the rest
Certain applications require specific network resources

Figure 5-1 End-to-end service models

For levels of service beyond basic IP services or best-effort service, signaling protocols and queuing, traffic shaping and filtering mechanisms are employed to supplement IP and provide the services required by the application. Keep in mind, however, that QoS is not a substitute for necessary bandwidth. Sufficient bandwidth is necessary, and more is certainly better, to minimize congestion within the network. When congestion does occur, QoS mechanisms help to ensure that less critical or more tolerant traffic will encounter network delay before application traffic with real-time requirements or that of a more important business nature.

Integrated or Reserved Services

With Integrated Services, or *IntServ,* a particular QoS is negotiated at the time it is requested. Resource Reservation Protocol (RSVP) is used to allow an application to request or signal the network to reserve a certain amount of bandwidth with particular QoS criteria, such as minimum latency. RSVP is defined in IETF Internet standard RFC 2205. RSVP can be used to provide something similar to a dedicated circuit over an IP network. See Figure 5-2 on page 104.

While RSVP provides the highest level of guaranteed services, it is also the most complex of the QoS protocols. This is because the reservation must be done across the entire data path with the state of each node maintained throughout the duration of the connection. This is certainly not a trivial task when thousands of reservation requests are anticipated. You can think of Integrated Services as a superset of the mechanisms necessary for the simpler *Differentiated Services*.

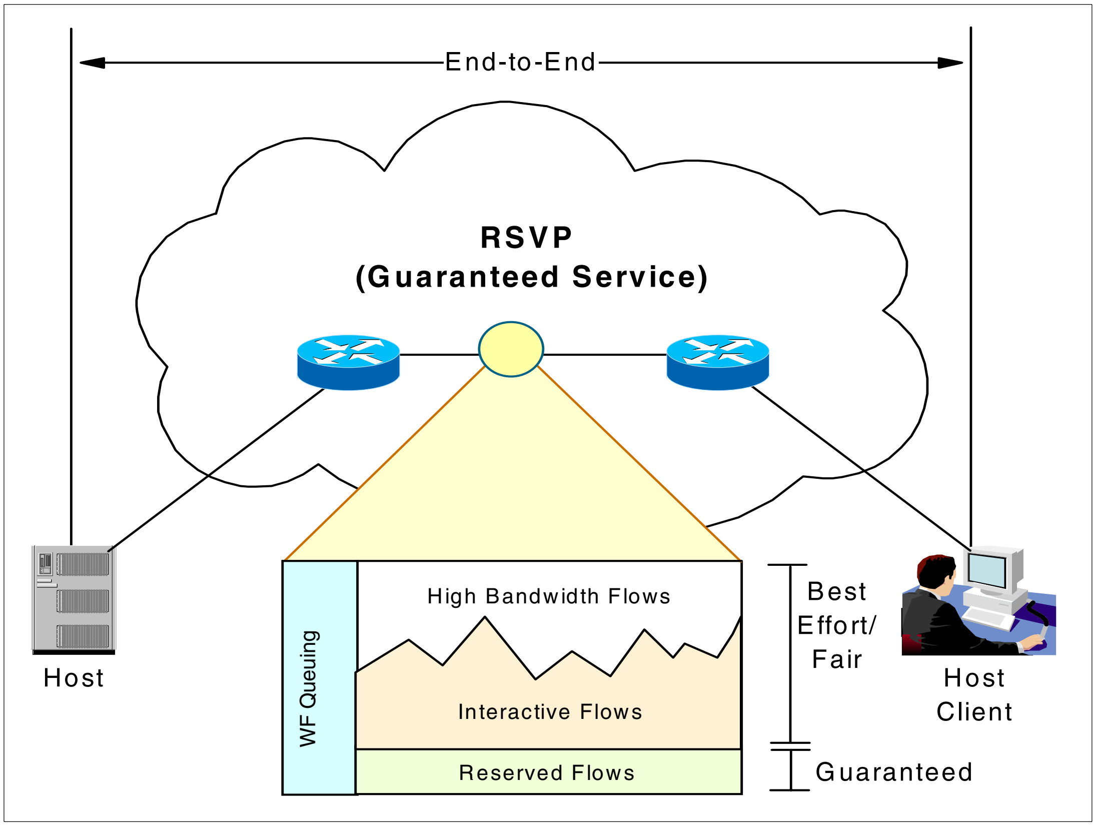


Figure 5-2 Integrated Services

A host uses RSVP to request a level of service on behalf of the application from the network before data is actually sent. Information is provided relative to the traffic profile the application expects to send and service is requested in terms of required bandwidth and maximum tolerated delay. The network responds to the QoS request based on available resources and then commits to meeting the requested service level or denies the request. Refer to Figure 5-3 on page 105.

Packet scheduler

The packet scheduler manages the forwarding of different packet streams in hosts and routers, based on their service class, using queue management and various scheduling algorithms.

Packet classifier

The packet classifier identifies packets of an IP flow in hosts and routers that will receive a certain level of service.

Admission control

The admission control contains the decision algorithm that a router uses to determine if there are enough routing resources to accept the requested QoS for a new flow.

The Resource Reservation Protocol (RSVP)

The Resource Reservation Protocol is used by Integrated Services to set up and control QoS reservations.

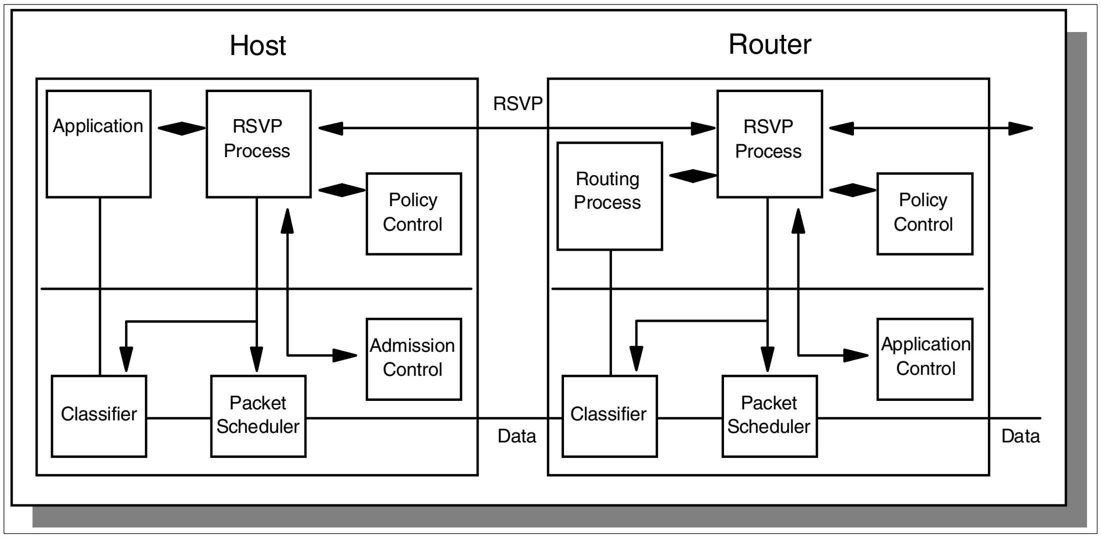


Figure 5-3 Integrated Services operational model

Service classes

The Integrated Services model includes two standard service classes defined by the IETF. Controlled Load Service is defined in RFC 2211 and the Guaranteed Service is defined in RFC 2212.

Controlled Load Service provides the data stream with approximately the same QoS as the stream would receive if the network were operating at its optimal uncongested state with best-effort service. It uses *admission control* to ensure this service is received even when the network becomes overloaded and packets begin to encounter queuing delays. The language, *same as uncongested,* is intentionally vague and meant to infer a rough approximation of an unloaded network's ability to deliver packets. A Controlled Load Service can expect to have a high percentage of packets delivered successfully and the delay experienced by a high percentage of packets will not be significantly more than the minimum transit delay of any one packet. Controlled Load Service is meant to be used for a wide range of applications, particularly those demanding applications that encounter problems when a network experiences increased delays above the uncongested condition.

Guaranteed Service, on the other hand, provides firm bounds on queuing delays and bandwidth capacity. The ability to provide the required service level is dependent on each node in the path supporting the service in an adequate fashion. Guaranteed Service ensures that packets will arrive at their destination within the guaranteed delivery time and will not be discarded due to queue overflows, provided the offered traffic load stays within the specified transmission specification. The service is intended for applications that require specific delay characteristics. Consider, for example, a video or audio stream where data can be buffered but beyond that point, the sound halts. The service controls merely the maximum queuing delay. Guaranteed Service provides approximately the same level of service as that provided by a *leased line*.

Differentiated Services

Differentiated Services (DiffServ) was developed to allow a network to support multiple service classes without the need to maintain the state of each traffic flow along the path or to perform signaling between nodes. It can, therefore, scale to support the traffic seen in today's global networks. The network domain manager or administrator defines aggregate traffic service classes, sometimes referred to as the Olympic classes: Platinum, gold, silver, and bronze. DiffServ is, therefore, less complex than Integrated or Reserved Services. It is less network intensive and is appropriate for networks of networks even where portions of the network are outside the control of the network domain manager.

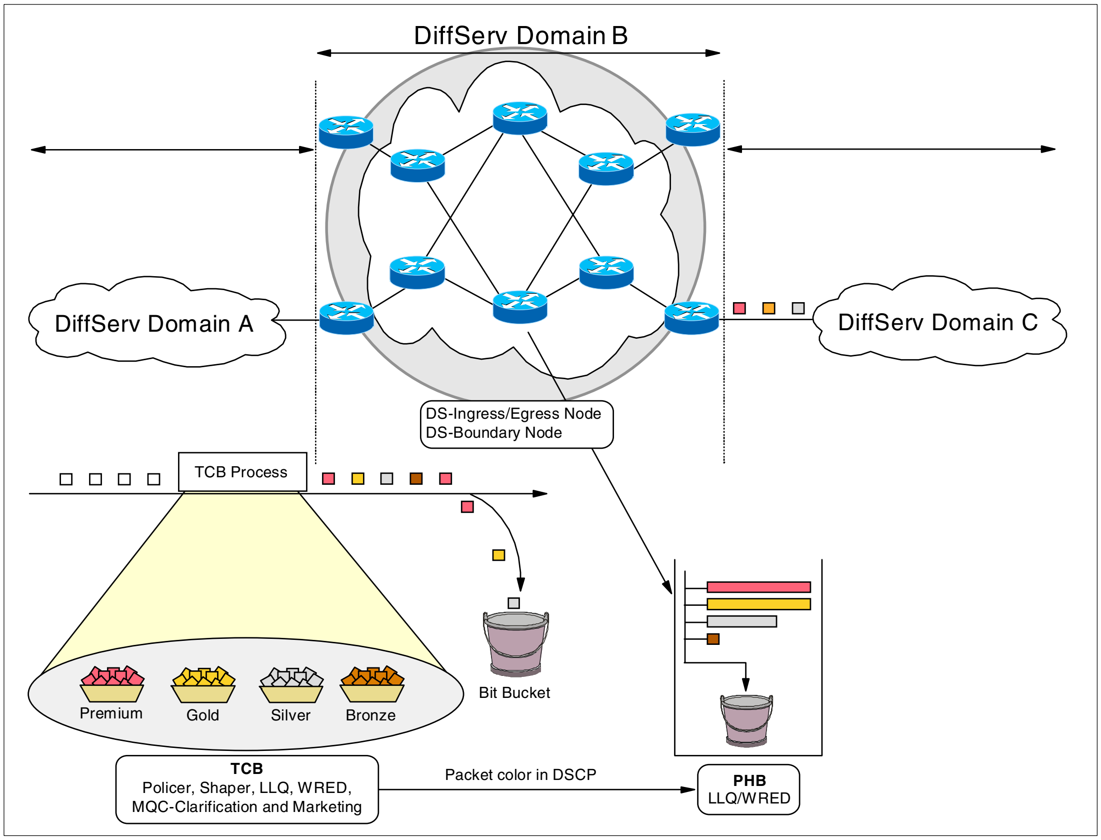


Figure 5-4 DiffServ end-to-end architecture

DiffServ is described in IETF RFC 2474, RFC 2475, RFC 2597, and RFC 2598. DiffServ is meant to handle traffic aggregates. This means that traffic is classified according to the application requirements relative to other application traffic. Each node then handles the traffic using internal mechanisms to control bandwidth, delay, jitter, and packet loss. Through the use of standard per-hop-behaviors (PHBs), packets receive the proper handling and the result is end-to-end QoS.

For true end-to-end QoS, each administrative domain must implement cooperative policies and PHBs. Packets entering a DiffServ domain can be metered, marked, shaped, or policed to implement traffic policies as defined by the administrative authority. This is handled by the DiffServ traffic conditioner block (TCB) function. DiffServ boundary nodes will typically perform traffic conditioning. A traffic conditioner typically classifies the incoming packets into predefined aggregate classes, meters them to determine compliance to traffic parameters, marks them appropriately by writing or re-writing the DSCP, and finally shapes the traffic as it leaves the node.

The DS field

To distinguish the data packets from different customers in DS-capable network devices, the IP packets are modified in a specific field. A small bit pattern, called the *DS field*, in each IP packet is used to mark the packets that receive a particular forwarding treatment at each network node. The DS field uses the space of the former TOS octet in the IPv4 IP header and the traffic class octet in the IPv6 header. All network traffic inside of a domain receives a service that depends on the traffic class that is specified in the DS field.

The DS field uses six bits to determine the Differentiated Services Code Point (DSCP) as defined in RFC 2474 and RFC 2475. This code point will be used by each node in the net to select the PHB. A two-bit currently unused (CU) field is reserved. The value of the CU bits are ignored by Differentiated Services-compliant nodes, when PHB is used for received packets. Figure 5-5 shows the structure of the defined DS field.

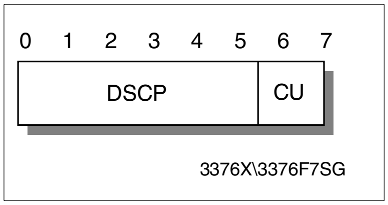


Figure 5-5 DS field

In the event that some nodes in a network recognize only the IP precedence bits, standard DSCP PHBs are constructed in such a way that they remain compatible with IP precedence. For example, the DSCP values can be used such that the values for IP precedence relate to the classes, as shown in Table 5-1.

Table 5-1 Relationship between IP precedence and DSCP

RFC 791 precedence		**RFC 2474, RFC 2475 DiffServ**	
Network Control	111 (7)	Preserved	111000
Internetwork Control	110 (6)	Preserved	110000
CRITIC/ECP	101 (5)	Express Forwarding	101xxx
Flash Override	100 (4)	Class 4	100xxx
Flash	011 (3)	Class 3	011xxx
Immediate	010 (2)	Class 2	010xxx
Priority	001 (1)	Class 1	001xxx
Routine	000 (0)	Best Effort	000000

5.2 Steps in QoS deployment

Network resources and bandwidth are of finite quantity. While there have been many predictions regarding free bandwidth, that is not the case today nor in the foreseeable future. Implementing QoS, then, means that in times of network congestion, some traffic will get better service while other traffic will likely see increased delay and lower throughput. Therefore, it makes sense to think of the network and applications as a system.

As a system, analysis and planning is necessary to ensure efficient operation. In order to successfully deploy QoS to meet the needs of an enterprise, a step-by-step process is followed prior to implementation. This includes the following steps:

- Traffic audit
- Traffic classification
- Defining policies
- Planning for RSVP configuration (if applicable)

5.2.1 Traffic audit

The first step in designing a network with QoS is to determine the different application requirements and allocate each traffic stream to the required class of service. For many organizations this is not as easy as it sounds. However, it is a critical step. This step can be broken down into the following tasks:

- Network audit

  A network audit determines what traffic is present in the network, the capacity required, and the time that it is required.

- Business audit

  Here the application traffic is ranked in terms of business importance.

- Application audit

  During the application audit, the specific network requirements and traffic characteristics of the applications are determined.

- Normalize ranks and assign service levels

  Once the audit process is complete, assignments can be made to correlate business and applications requirements with a set of service levels. It may be necessary to revisit the current network design, since this exercise may have identified requirements beyond what the network is capable of delivering.

5.2.2 Traffic classification

Using the results of the auditing step, specific application traffic or traffic categories are assigned to a number of classes. We suggest you begin with a small number of classes, perhaps using the Olympic reference model of platinum, gold, silver, and bronze classifications. Figure 5-6 on page 109 illustrates the Olympic scheme.

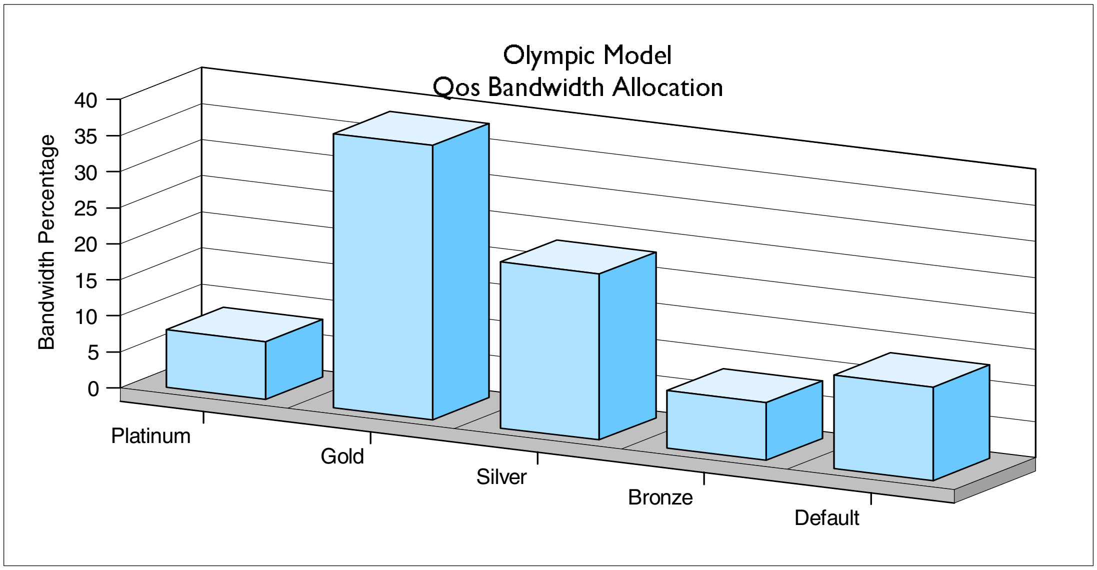


Figure 5-6 Olympic scheme

There are many ways that a given traffic stream can be classified. A class can be assigned by application using such identifiers as Uniform Resource Locators (URLs), IP ports, tasks, or application name. Classes can also be assigned based on network criteria such as origin or destination IP address, MAC address, or a combination. And groups or departments could be assigned a class based on their IP subnet or network interface, for example. The point is that there are so many ways classes can be assigned that restraint is necessary to minimize the number of classes and assignment mechanisms so complexity is mitigated.

Two other points are worth mentioning. First, we recommend that you classify, or *color*, traffic nearest the source at the edge of the network by setting the DSCP bits to properly designated values. Avoid host, application-based coloring of traffic when possible, since this will inevitably become a problem if not managed properly and may create inconsistencies between traffic streams.

5.2.3 Defining policies for the classes

When defining policies for a class, you are defining the actual end-to-end service you want the traffic stream assigned to the class to receive. We are getting down to detail now. Here we are specifying service in terms such as minimum guaranteed bandwidth, the maximum amount of bandwidth this class will be permitted to use, and assigning a priority in relation to other traffic classes.

Important: The goal of your policy definitions, when considered as a system, is to offer an appropriate combination of consistent services that meet the requirements of all traffic streams.

Take network topology into account and plan for future change and growth. Also document all policies and classes and the desired relationships between traffic categories.

5.2.4 Planning for RSVP configuration

Bandwidth reservation using RSVP signaling requires careful planning. First make sure reservation is necessary. DiffServ may be sufficient and simpler to implement. Often data traffic does not need reserved bandwidth. Only certain applications with particular real-time requirements that cannot be accommodated using DiffServ techniques will use reservation signaling. Usually, these are applications that are multicast in nature. However, RSVP can be used for unicast between two application endpoints.

Consider these questions when planning for RSVP:

- How much bandwidth should be allowed per application flow?
- How much bandwidth must be excluded from RSVP reservations so that normal traffic is serviced properly?

It is imperative that you understand the traffic mix, network performance characteristics, and application requirements before entering RSVP configuration commands that affect network traffic.

5.3 QoS on the z/OS Communications Server

In the z/OS CS environment, support for Integrated Services is provided by the RSVP Agent. The RSVP Agent queries the Policy Agent for relevant information and communicates with the Cisco router to request the desired QoS on behalf of the application.

Differentiated Services is supported by the z/OS CS Policy Agent (PAGENT). PAGENT reads policy definitions from a local configuration file or a Lightweight Directory Access Protocol (LDAP) server. PAGENT then installs the policies in the z/OS CS stacks as desired. Support for environments with multiple TCP/IP stacks is possible using the configuration techniques described in 5.3.2, “Configuring QoS in z/OS Communication Server” on page 114.

Figure 5-7 on page 111 shows the relationship between the various z/OS QoS components. Tasks or daemons such as PAGENT and RSVPD work together and with the TCP/IP protocol stack to classify and mark packets for QoS. Data collection points are also available for performance management.

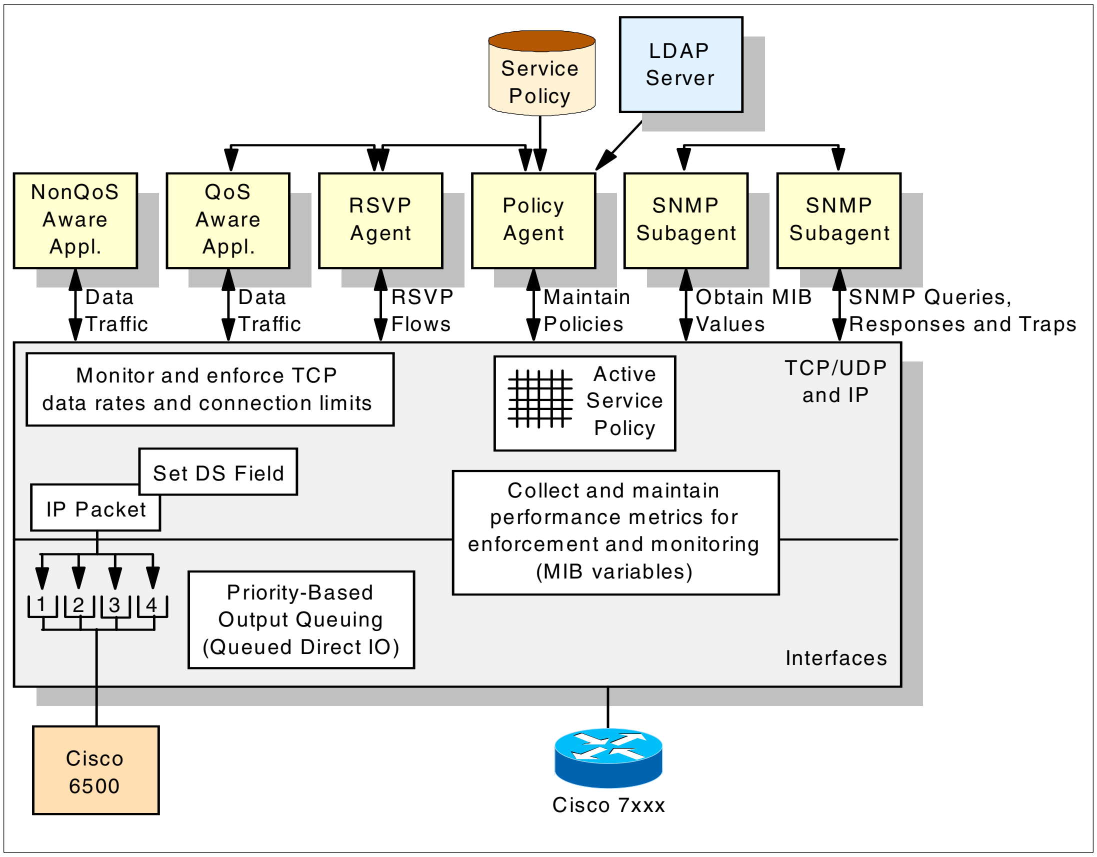


Figure 5-7 z/OS CS QoS components

A cooperative framework of host-based components and QoS mechanisms within the Cisco network allow for end-to-end service levels to be established.

5.3.1 PAGENT policies

We suggest that when you first implement QoS policies you start with a small number of critical applications or traffic types. Then, as you develop more knowledge of the traffic patterns and interactions, continue to apply a set of service classes to applications or traffic streams as needed.

The Policy Agent (PAGENT) supports the following types of policies:

- Quality of Service (QoS) policies
  - Differentiated Services (DS) policies
  - Integrated Services (RSVP) policies
  - Sysplex Distributor (SD) policies
- Intrusion Detection Services (IDS) policies
  - Scan policies
  - Attack policies
  - Traffic Regulation policies

QoS policies

Policy conditions consist of a variety of selection criteria that act as traffic filters. Traffic can be filtered based on source/destination IP addresses, source/destination ports, protocol, inbound/outbound interfaces, application name, application-specific data, or application priority. Only packets that match the filter criteria are selected to receive the accompanying action. Policy rules can refer to several policy actions, but only one policy action is executed per policy scope. A given policy action may be referred to by several policy rules.

Differentiated Services (DS) policies

Policies to be implemented can be configured via the Policy Agent configuration file, in an LDAP Server, or both. Once read, the policies are combined into a single list. Policy rules and actions map subsets of outbound traffic to various QoS classes and can be used to create end-to-end Differentiated Services.

Setting DSCP using the Policy Agent

PAGENT policies are defined by rules and actions. The rules consists of a variety of selection criteria to provide a *match condition*. Matching the *rule* then forces the *action*.

The following actions can be performed using Differentiated Services policies:

- Set the DS field or Type of Service (TOS) byte and map to S/390 Queued Direct I/O (QDIO) device priority
- Committed access bandwidth (mean rate and peak rate) control and enforcement
- TCP connection limits
- Maximum and minimum TCP connection rates, TCP maximum delay

Of particular importance here is the setting of the DS field. Outbound traffic can be marked with the desired Differentiated Services Control Point (DSCP) value. This marking will then be interrogated by the Cisco router or switch and the appropriate PHB applied as the packet traverses the network.

The host PAGENT can be defined as a started task. Upon startup it reads a configuration data set that contains the commands to configure the Policy Agent.

Integrated Services (RSVP) policies

Although RSVP policies are installed into the TCP/IP stack, they are only used for collecting policy statistics. For policy use and limit enforcement, these policies are requested from the Policy Agent by the RSVP Agent, to apply to RSVP reservation requests from RSVP applications.

RSVP Agent

The RSVP Agent includes an application programming interface called RAPI. The RAPI interface is a set of C language routines that allow a custom application to make enhanced QoS requests. The RSVP Agent then issues RSVP protocol messages to the network. Each router in the path of the data flow may accept or deny the request, depending on the resources available to meet the requirement. If the request is denied, the RSVP Agent returns the decision to the application using the RAPI.

Applications can use the RSVP Agent to establish resource reservations within the network. Reservation requests include a Traffic Specification, or Tspec, that consists of the following values:

- Token bucket mean rate (r)
- Token bucket depth (b)
- Peak rate (p)

- Minimum policed unit (m)
- Maximum packet size (M)

The RSVP Agent can be defined as a started task. Upon startup it reads a configuration data set that contains the commands necessary to configure the RSVP Agent. Applications invoke QoS reservations using the RSVP application programming interface (RAPI). Information on RAPI can be found in *z/OS V1R2 Communications Server: IP Application Programming Interface Guide*, SC31-8788.

Sysplex Distributor policies

Sysplex Distributor policies are used to specify a group of target nodes for a given traffic set. For example, suppose you want all FTP traffic from a certain subnet to be assigned one group of systems and FTP traffic from another subnet to be assigned to a different set of systems. This can be accomplished using SD policies.

SD policies establish load balancing rules by combining Workload Manager (WLM) information with the defined SD policy.

The goal of SD policy is to limit the target TCP/IP stacks for inbound traffic from a given subnet. The Policy Agent running on SD target nodes within a sysplex can collect network QoS performance data on behalf of policies defined for a target port or application. It then assigns a weight fraction to such a policy. This weight is then used by the SD distributing stack, in conjunction with weights assigned by the Workload Manager, to take QoS performance into consideration for load balancing decisions. The PolicyPerfMonitorForSDR statement in the PAGENT configuration file is used to define the characteristics of the SD policy performance monitoring. Figure 5-8 on page 114 illustrates the relationship of SD policies, the distributing stack, and the target stacks.

The SD policies should be installed in the SD distributing stack, while the SD policy performance monitoring is done by the Policy Agent running on the SD target stacks.

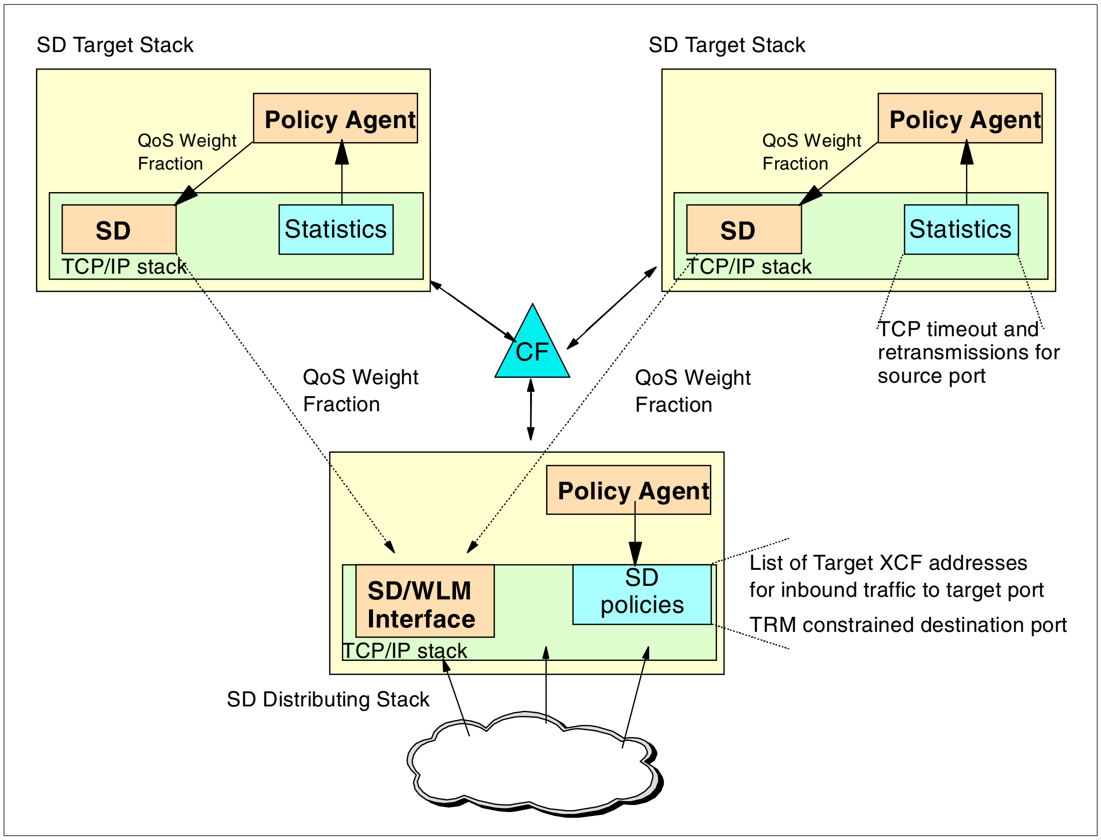


Figure 5-8 Sysplex Distributor policies

Intrusion Detection Services

Intrusion Detection Services (IDS) support is available to detect and report on network intrusion events. The Traffic Regulation and Management (TRM) support provided in V2R10 has been extended and incorporated into the IDS support. IDS policy regulates the types of events to report and provides the definition of several types of events. IDS policy may be defined for scans, attacks, and Traffic Regulation for both TCP and UDP ports.

Notes: IDS policies may be defined only on an LDAP Server.

Policy Scope TR policies found in a Policy Agent configuration file are compatibly transformed into IDS TR TCP policies by the Policy Agent.

`pasearch` will display the transformed policy.

In this chapter, we are concerned with QoS and therefore do not cover security and intrusion detection policies.

5.3.2 Configuring QoS in z/OS Communication Server

The two components mainly responsible for QoS within z/OS CS are the Policy Agent and the RSVP Agent. In this section, we provide an overview of the configuration steps necessary to use the z/OS CS Policy Agent.

PAGENT supports QoS functions other than reading/installing policies, such as Sysplex Distributor policy performance monitoring, and mapping Type of Service (TOS) byte values to outbound interface and virtual LAN (VLAN) user priorities.

The Policy Agent runs in the z/OS environment and reads policy definitions from a local configuration file and/or a central repository that uses the Lightweight Directory Access Protocol (LDAP). The Policy Agent also installs policies in one or more z/OS CS stacks. It can be used to replace existing policies or update them as necessary.

The z/OS CS RSVP Agent provides Integrated Services functions, such as communicating with RSVP Agents on other hosts/routers and reserving resources on certain types of outbound interfaces. The RSVP Agent queries the Policy Agent for policies that relate to RSVP processing.

Basic configuration

Before defining policies, some basic operational characteristics of the Policy Agent need to be configured in the PAGENT configuration file. In this section, we detail the following configuration steps:

1. Define the TCPImage statements.
2. Define the appropriate logging level.
3. Define security product authorization for the Policy Agent.

Define the TcpImage statement(s)

The Policy Agent can be configured to install policies on one or more TCP/IP stacks, or images. Each TCP/IP stack is configured using a TcpImage statement in the main configuration file. A secondary configuration file can be defined for any given stack, a set of stacks can share configuration information in the main configuration file, or a combination of these techniques can be used.

To install different sets of policies to different stacks, configure each image with a different secondary configuration file. In this case, each image can be configured with a different policy refresh interval if desired. The refresh interval used for the main configuration file will be the smallest of the values specified for the different stacks.

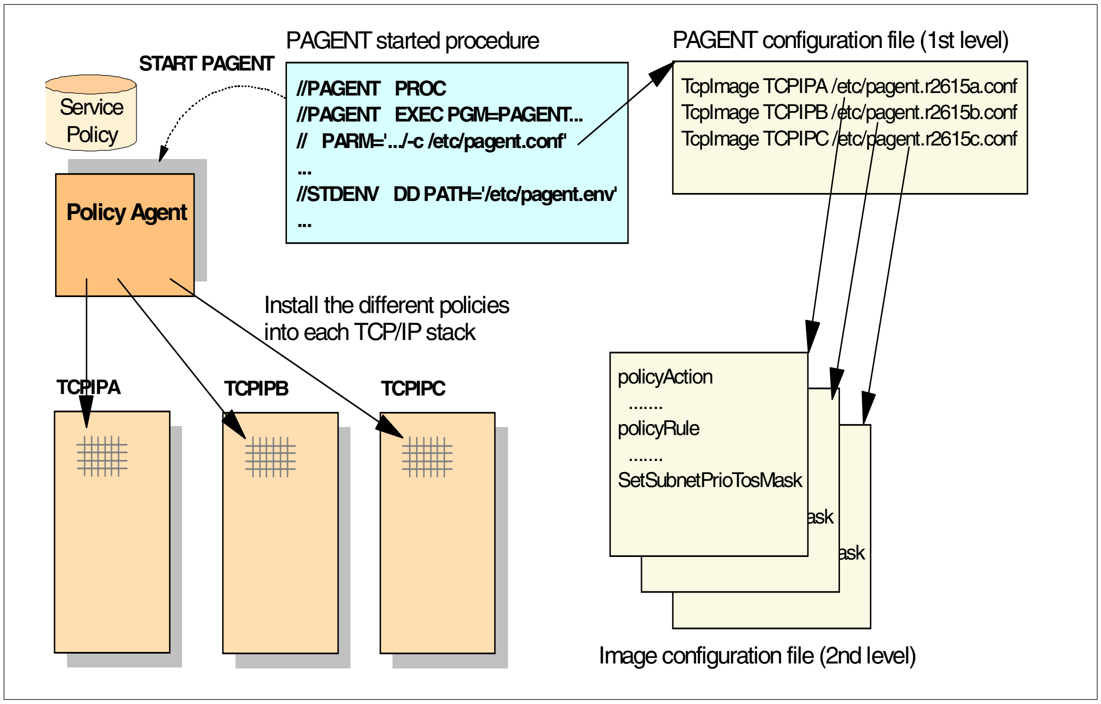

PAGENT started procedure
START PAGENT
//PAGENT PROC
//PAGENT EXEC PGM=PAGENT...
// PARM='../-c /etc/pagent.conf'
...
//STDENV DD PATH='/etc/pagent.env'
...
PAGENT configuration file (1st level)
TcpImage TCPIPA /etc/pagent.r2615a.conf
TcpImage TCPIPB /etc/pagent.r2615b.conf
TcpImage TCPIPC /etc/pagent.r2615c.conf
Service Policy
Policy Agent
Install the different policies into each TCP/IP stack
TCPIPA
TCPIPB
TCPIPC
policyAction
....
policyRule
....
SetSubnetPrioTosMask
ask
ask
Image configuration file (2nd level)

Figure 5-9 Multiple stacks, multiple policy definitions

Note: When the main configuration file is an MVS data set, it is reread at each refresh interval (which is the smallest of the individual stack refresh intervals), regardless of whether it has actually been changed. Because PAGENT restarts all stack-related processing when the main configuration file is reread, this effectively makes the refresh interval for all stacks the same as this smallest configured interval.

To install a common set of policies to a set of stacks, do not specify secondary configuration files for each image. In this case, there is only one configuration file (the main one) and the policy information contained in it is installed to all of the configured stacks. Different refresh intervals can also be configured for each image, but would probably be less useful in this case.

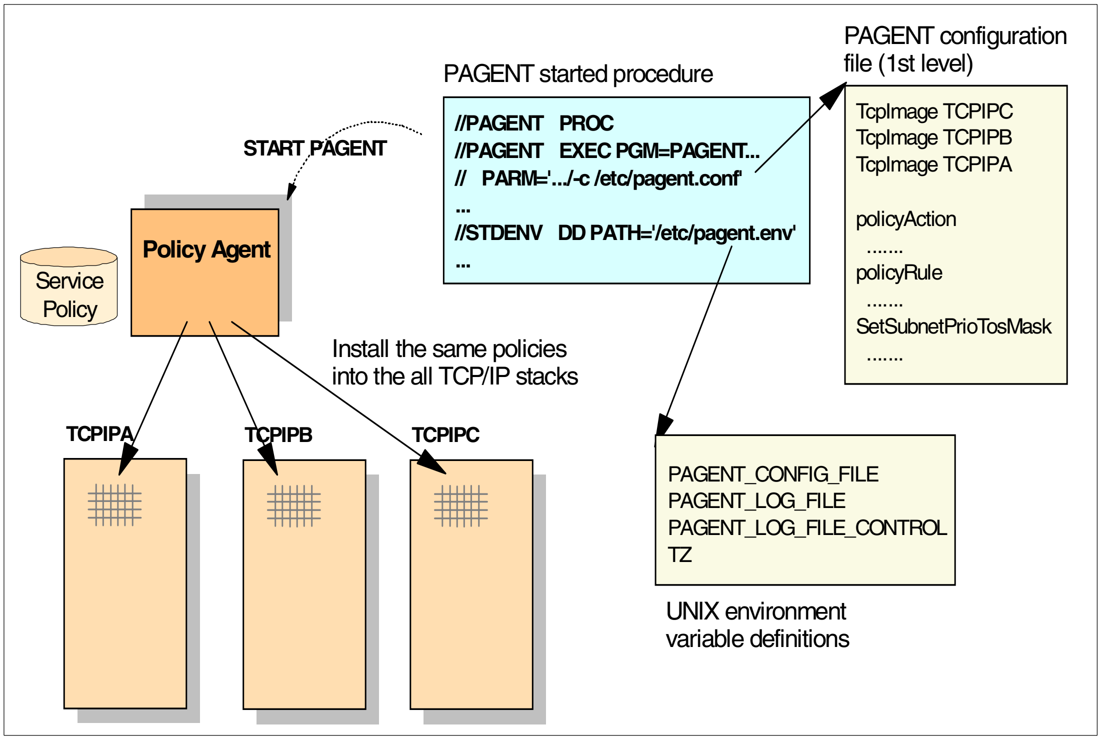


Figure 5-10 Multi-stack, single policy definition

In either case, it is possible that TCP/IP stacks configured to the Policy Agent are not started or even defined. The Policy Agent will fail when trying to connect to those stacks and log appropriate error messages.

The Policy Agent does not end when any (or all) stacks end. When the stacks are restarted, active policies are automatically reinstalled.

When the Policy Agent is shut down normally (that is, using KILL or STOP), then if the TcpImage statement option PURGE was coded, all policies will be purged from this stack.

The TcpImage statement specifies a TCP/IP image and its associated configuration file to be installed to that image. The following example installs the policy control file /tmp/TCPCS.policy to the TCPCS TCP/IP image, after flushing the existing policy control data:

```
TcpImage TCPCS /tmp/TCPCS.policy FLUSH
```

Define the appropriate logging level

The LogLevel statement is used to define the amount of information to be logged by the Policy Agent. The default is to log only event, error, console, and warning messages. This might be appropriate for a stable policy configuration, but more information might be required to understand policy processing or debug problems when first setting up policies or when making significant changes. Specify the LogLevel statement with the appropriate logging level in the main configuration file.

Note: The maximum logging level (511) can produce a significant amount of output, especially with large LDAP configurations. This is not a concern if an HFS log file is used, because the Policy Agent uses a set of log files with a finite size in a round-robin (circular) configuration (the number and size of these files is controllable with the PAGENT_LOG_FILE_CONTROL environment variable). But when using the syslog daemon as the log file, the amount of log output produced should be taken into consideration.

Define security product authorization for the Policy Agent

Because the Policy Agent can affect system operation significantly, security product authority (for example, RACF) is required to start the Policy Agent. Refer to the EZARACF sample in SEZAINST for sample commands needed to create the profile name and permit users to it.

If Policy Agent clients (that is, pasearch) are *not* defined as a superuser, then security product authority in the SERVAUTH CLASS for that client *must* be defined to retrieve policies. These profiles can be defined by TCP/IP stack (that is, TcpImage) and policy type (that is, ptype = QoS or IDS). Wildcarding of profile names are allowed.

```
EZB.PAGENT.sysname.TcpImage.ptype
```

Where:

- *sysname* is the system name defined in sysplex.
- *TcpImage* is the Tcp name for policy information being requested.
- *ptype* is the policy type that is being requested, where:
  - QOS is the Policy QoS.
  - IDS is the Policy IDS.

Note: Wildcarding is allowed on segments of the profile name.

The Policy Agent will check all client's requests to verify that SERVAUTH class is active and the profile exists for the TcpImages and policy types in the request. If a client's request is for *all* TcpImages and policy types defined, then the Policy Agent will only return information for any object for which permission is granted. For example, if the request is for *all* policy types, and both QoS and IDS policy types are defined, but the user is only granted permission for the QoS policy types, then only QoS policy information will be returned.

If SERVAUTH class is absent (not RACLIST), or profiles are absent for a client's request (that is, TcpImage, and policy type), then permission is denied and data is *not* returned.

If SERVAUTH class is active and profiles are present for a client's request (that is, TcpImage and policy type) and an MVS user is defined for all profiles, then permission is granted and data is returned.

If SERVAUTH class is active and profiles are present for a client's request (that is, TcpImage and policy type) and an MVS user (Policy Agent client) is not defined for all profiles, then permission is refused and data is not returned.

Refer to the EZARACF sample in SEZAINST for sample commands needed to create the profile name and permit users to it.

Defining policies

Policies consist of several related objects. The main object is the *policy rule.* A policy rule object refers to one or more *policy condition*, *policy action*, or *policy time period condition objects*, and also contains information on how these objects are to be used. Policy time period objects are used to determine when a given policy rule is active. Active policy objects are related in a way that is analogous to an IF statement in a program. For example:

```
IF condition THEN action
```

In other words, when the set of conditions referred to by a policy rule are TRUE, then the policy actions associated with the policy rule are executed.

Differentiated Services rule

The most common QoS deployment will use rules to map outbound traffic from particular applications into sub-classes. Example 5-1 illustrates this type of policy. The goal of this Differentiated Services policy is to map a subset of the traffic outbound from an FTP server.

Example 5-1 Sample DiffServ rule

```
PolicyRule    diffServ
{
  ProtocolNumberRange                 6
  SourceAddressRange                  200.50.23.11
  SourcePortRange                     20-21
  PolicyActionReference               tokenbucket
  PolicyRulePriority                  10
  ConditionTimeRange                  20000701000000:20050630235959
  DayOfMonthMask                      1111111111111111111111111111111
  DayOfWeekMask                       0111110
  TimeOfDayRange                      06:00-22:00
}
PolicyAction  tokenbucket
{
  PolicyScope                         DataTraffic
  OutgoingTOS                         10000000
  DiffServInProfileRate               256      # 256 Kbps
  DiffServInProfileTokenBucket        512      # 512 Kbits
  DiffServInProfilePeakRate           512      # 512 Kbps
  DiffServInProfileMaxPacketSize      120      # 120 Kbits
  DiffServOutProfileTransmittedTOSByte 00000000
  DiffServExcessTrafficTreatment      BestEffort
}
```

This policy is identified as a Differentiated Services policy by the PolicyScope DataTraffic attribute on the PolicyAction statement, as well as the use of several DS-only attributes.

The following statements apply to the example in this section:

- The policy rule selects traffic originated by ports in the range 20–21 for TCP (FTP outbound data connection uses port 20) from the source address 200.50.23.11.
- The policy rule is active on weekdays between 6 a.m. and 10 p.m. local time, between the dates 7/1/2000 and 7/1/2005.
- The policy action specifies that the TOS byte be set to '10000000' for traffic that conforms to this policy.
- The action establishes a *token bucket* traffic conditioner with a mean rate of 256 kilobits per second, a peak rate of 512 kilobits per second, and a burst size of 64 kilobytes. Any

traffic that exceeds these specifications will be sent as *best effort*, with an accompanying TOS byte of '00000000'.

Example 5-2 shows another example of a DS policy.

Example 5-2 Web policy sample

```
PolicyRule web-catalog       # web catalog traffic
{
   protocolNumberRange  6
   SourcePortRange    80
   ApplicationData /catalog
   policyActionReference  interactive1
}
PolicyAction interactive1
{
   policyScope DataTraffic
   outgoingTOS 10000000
}
```

The goal of this policy is to ensure that outgoing data that matches the specified attributes will be assigned a QoS service level defined in action "interactive1".

The following statements apply to Example 5-2 in this section:

- This rule will only match traffic on TCP connections (protocol 6) with a source port of 80 (that is, HTTP server) and application-defined data beginning with the string "/catalog".
- Since we are dealing with HTTP traffic, this rule is basically indicating that all outgoing traffic associated with a URI that begins with "/catalog" should be managed using the DS characteristics specified in the "interactive1" policy action.

RSVP policy example

The goal of this RSVP policy is to establish limits on resource reservations requested by RSVP applications using the RSVP API (RAPI) interface. The policy is identified as an RSVP policy by the PolicyScope attribute on the PolicyAction statement, as well as the use of RSVP-only attributes.

Example 5-3 RSVP policy sample

```
PolicyRule    intserv
{
  SourcePortRange                       8000 8001
  ProtocolNumberRange                   6
  PolicyActionReference                 intserv1
  PolicyActionReference                 intserv2
}
PolicyAction  intserv1
{
  PolicyScope                           DataTraffic
  OutgoingTOS                           01100000
}
PolicyAction  intserv2
{
  PolicyScope                           RSVP
  OutgoingTOS                           01100000
  FlowServiceType                       ControlledLoad
  MaxRatePerFlow                        400    # 50000 bytes/second
  MaxTokenBucketPerFlow                 48     # 6000  bytes
  MaxFlows                              10
}
```

The following statements apply to Example 5-3 on page 120.

- The policy rule selects traffic from source ports in the range 8000 to 8001, with a protocol ID of 6 (TCP).
- The DataTraffic policy action specifies that the TOS byte be set to '01100000' for Differentiated Services traffic that conforms to this policy. Essentially, any traffic sent by the target application without an RSVP reservation in place will use this policy action. Once an RSVP reservation is in place, the RSVP action gets used.
- The RSVP policy action specifies that the TOS byte be set to '01100000' while an RSVP reservation is in place. It also limits the type of RSVP service requested by RSVP applications to Controlled Load. Applications requesting Guaranteed Service are "downgraded" to using Controlled Load Service. In addition, the action limits the mean rate and token bucket size to 50000 bytes per second and 6000 bytes, respectively. These values are requested by RSVP applications in the traffic specification, or Tspec.
- The action also limits the number of active RSVP flows that map to this policy to 10.

For complete information about defining policies and configuring the PAGENT, refer to *z/OS V1R2 Communications Server: IP Configuration Reference*, SC31-8776, and *z/OS V1R2 Communications Server: IP Configuration Guide*, SC31-8775.

Configuring the RSVP Agent

To configure the RSVP Agent, update the configuration file to specify RSVP Agent operational parameters using the LogLevel, TcpImage, Interface, and RSVP statements.

To configure the RSVP Agent, you must first authorize the RSVP Agent using the security product. See SEZAINST(EZARACF) for SAF considerations for started tasks. Example 5-4 shows a sample RSVP configuration file.

Example 5-4 Sample RSVP configuration file

```
Interface 10.11.12.13 Disabled
{}
Interface 200.1.1.1 Enabled
{
  TrafficControl Disabled
}
Interface Others Enabled
{}
Rsvp All Enabled
{
  MaxFlows 50
}
```

This example:

- Runs the RSVP Agent on the stack selected using the standard resolver search order, because a TcpImage statement is not configured.
- Disables RSVP processing on interface 10.11.12.13, while enabling it for all other interfaces.
- Disables traffic control on interface 200.1.1.1. This means that no reservations will be made on this interface.
- Allows a maximum of 50 active RSVP flows per interface.

For complete information on how to configure the RSVP Agent, refer to *z/OS V1R2 Communications Server: IP Configuration Reference*, SC31-8776, and *z/OS V1R2 Communications Server: IP Configuration Guide*, SC31-8775.

5.4 Managing Quality of Service

So far we have discussed QoS in terms of how to enable the network to handle various applications with different network requirements. But what network remains static? Equally important is the ability to manage the QoS features and services as well as constantly reviewing policies for network resource allocation. QoS essentially provides better service to some and worse service to others, with the goal being, at least, adequate service to all. QoS management helps to keep it that way and provide assurances that the network is functioning as designed, that the service levels are what the architects intended, and that service levels are what the customers require and are entitled to.

5.4.1 Available management tools

There are several tools available that will help you manage your network QoS configuration and parameters.

z/OS CS SNMP SLA Subagent

The z/OS CS SLA Subagent allows network administrators to retrieve data and determine if the current set of SLA policy definitions are performing as needed or if adjustments need to be made. The SLA Subagent supports the Service Level Agreement Performance Monitor (SLAPM) MIB. Refer to RFC 2758 for more information about the SLAPM MIB.

Cisco QoS MIBs

The new Cisco Class-Based QoS MIB provides you with the same statistics that the **`show policy interface`** command provides.

To measure packet loss through the network with regard to different classes of service, use Class-Based Queuing over Security Management Information Base [CBQoSMIB (available in 12.1(5) T], SAA/IPM, or QPM.

See also:

- CISCO-CLASS-BASED-QOS-MIB.my
- CISCO-CLASS-BASED-QOS-MIB-CAPABILITY.my

Cisco QoS Device Manager

Cisco QoS Device Manager (QDM) is a Web-based network management application that provides an easy-to-use graphical user interface for configuring and monitoring advanced IP-based Quality of Service functionality in Cisco Systems routers.

QDM is intended for users that are configuring QoS functionality in their network for the first time. These customers need an easy-to-use management application to help them configure and monitor QoS features in the most critical router devices in their networks. Using QDM, network managers can quickly and easily configure QoS functionality and immediately observe the effect that this QoS configuration has on the pattern of network traffic through the router.

Cisco QoS Policy Manager

QPM is a QoS policy system that makes it easy to define traffic policies and automate multiple service levels across any network topology. The product enables network-wide, content-based Differentiated Services, centralized policy control for voice/video/data networks, automated QoS configuration and deployment, and campus- to-WAN policy control. By automating the process of translating application performance requirements into QoS policy, QPM helps ensure reliable performance for Internet business applications and voice traffic that contends with noncritical traffic. Using QPM, a network administrator can quickly construct rules-based QoS policies that identify and partition application traffic into multiple levels of service.

5.5 VLAN Priority Tagging

QoS support, in PAGENT, allows the Type of Service (TOS) byte, also known as the Differentiated Services (DS) field to be set in the outbound IP packet header via service policies. On z/OS, the outbound TOS/DS value can be mapped to QDIO device priorities with the SetSubnetPrioTosMask statement or a LDAP object. As most LAN media have no inherent ability to propagate user priority in the MAC header, user-specified packet priorities may not be propagated to directly attached LAN devices (bridges and switches). Thus, the TOS byte is the mechanism for propagating packet priority.

In CS for z/OS V1R2, we implement a portion of IEEE standard 802.1Q that supports what is known as tagging of Virtual LAN frames, to allow the propagation of packet priority in LAN frames. A VLAN is defined as a subset of the active topology of a Bridged Local Area Network. The user priority field itself is specified in three header bits capable of representing eight priority levels.

The SubNetPrioTosMask statement/LDAP object has been extended to allow the TOS/DS value to be mapped to one of the 8 VLAN user priority values as shown in Figure 5-11.

```
SetSubnetPrioTosMask
{
     SubnetAddr 9.100.1.12
     SubnetTosMask 11100000
     PriorityTosMapping 1 11100000 7
     PriorityTosMapping 1 11000000 6
     PriorityTosMapping 1 10100000 5
     PriorityTosMapping 1 10000000 4
     PriorityTosMapping 2 01100000 3
     PriorityTosMapping 3 01000000 2
     PriorityTosMapping 4 00100000 1
     PriorityTosMapping 4 00000000 1
}
```

Figure 5-11 QDIO priority to VLAN priority mapping

The format of PriorityTosMapping is *priority tos user_priority*.

The first priority is the QDIO priority (1–4; 1 is the highest).

The second priority is the VLAN priority (0–7; 7 is the highest).

The default user priority is 0.

Note: The minimum hardware required to implement VLAN Priority Tagging is OSA-Express (QDIO) on a z900 hardware series with Driver level 3C.

5.6 QoS summary

Network requirements are driven by application requirements. While that statement is true, often the capabilities of the network lead to new application development that was not previously possible. The result is constant change and continual advancement of the capabilities of both applications and networks leading to higher and higher productivity levels.

As application data transfer requirements shift from the store-and-forward variety to more real-time streaming and multimedia needs, the ability to tailor network services more closely to the actual application requirements is increasingly important.

Enterprises need QoS to deploy a converged network utility that supports Voice over IP, multimedia e-learning, video conferencing, Web-based services, transaction-based applications, and the list goes on.

Service providers benefit in much the same way by passing on the same services to their customers. In addition, QoS allows providers to offer tiered services, QoS-based service level agreements (SLAs), and on-demand services such as content streaming, video conferencing, etc.

5.6.1 QoS reduces costs

Deploying QoS within the network and cooperatively based on application requirements reduces the total cost of communications services in many ways.

- QoS reduces wide area network costs by using available bandwidth more efficiently.
- The combination of voice and data over the same network reduces long-distance fees and presents a more cost-effective alternative to inter-office trunks.
- With service policies in place, network managers can concern themselves more with managing the traffic mix than chasing down each performance problem, application by application.

6

QoS configuration using the zQoS Manager

This chapter serves as a pseudo user's guide for the zQoS Manager tool. The zQoS Manager is a tool designed to allow a network administrator to produce the following:

- A file the LDAP Server can process (using either LDAP Version 2 or 3)
- A configuration file for the Policy Agent (PAGENT)

This chapter contains the following sections:

6.1 What zQoS is

zQoS Manager enables centralized configuration of Quality of Service policies for z/OS V1R2 using an LDAP Server as the policy repository. It provides a user-friendly interface with help panels to isolate network administrators from having to know the LDAP Policy Schema and the complexity of directly writing to the LDAP Server.

The zQos Manager is a product designed to allow a network administrator to produce the following:

- A file the LDAP Server can process (using either LDAP Version 2 or 3)
- A basic configuration file for the Policy Agent (PAGENT) that identifies the LDAP Server

Note: The zQoS Manager does not produce an output file that contains policies in the format that can be used in the configuration file.

The zQoS Manger is built with a Graphical User Interface (GUI) that provides a user-friendly front end for the entry of policy information. The policies (possibly incomplete) entered via the GUI may be stored on your local machine as an XML or LDIF file. You can then use the tool to read in and modify the XML file with updated policies. You can save, load, and change this file repeatedly to update policies for one or more policy agent configurations.

Note: The zQoS Manager can both read and write an XML file. However, it can only write (or create) an LDIF file.

The QoS policies used by PAGENT can either be stored in the PAGENT configuration file or an LDAP Server. If using the LDAP Server, the policies are required to be coded in accordance with RFC 2849 and as per the RFC, stored as an LDIF file. The zQoS Manager does produce a configuration file with the LDAP Server information required by PAGENT. This file can be sent via FTP or moved to and placed in the PAGENT configuration file. Given this, there are two methods for generating the LDIF file:

- Manually coding the policies (LDIF) file, and transferring the information to the LDAP Server
- Using the zQoS Manager to generate the LDIF file and configuring the zQoS Manager to automatically send the information to the LDAP Server

If you intend to store your QoS policies in an LDAP Server, we at the ITSO strongly recommend using the zQoS Manager tool to generate and transfer the policy information. As can be seen in 4.3.1, “Expressing the policies to LDAP” on page 83, the quantity of information needed for a single policy is quite substantial and the time saved by using the zQoS Manager to generate the LDIF file is significant. Figure 6-1 on page 127 shows the communication flow between the zQoS Manager, the LDAP Server, and the PAGENT application.

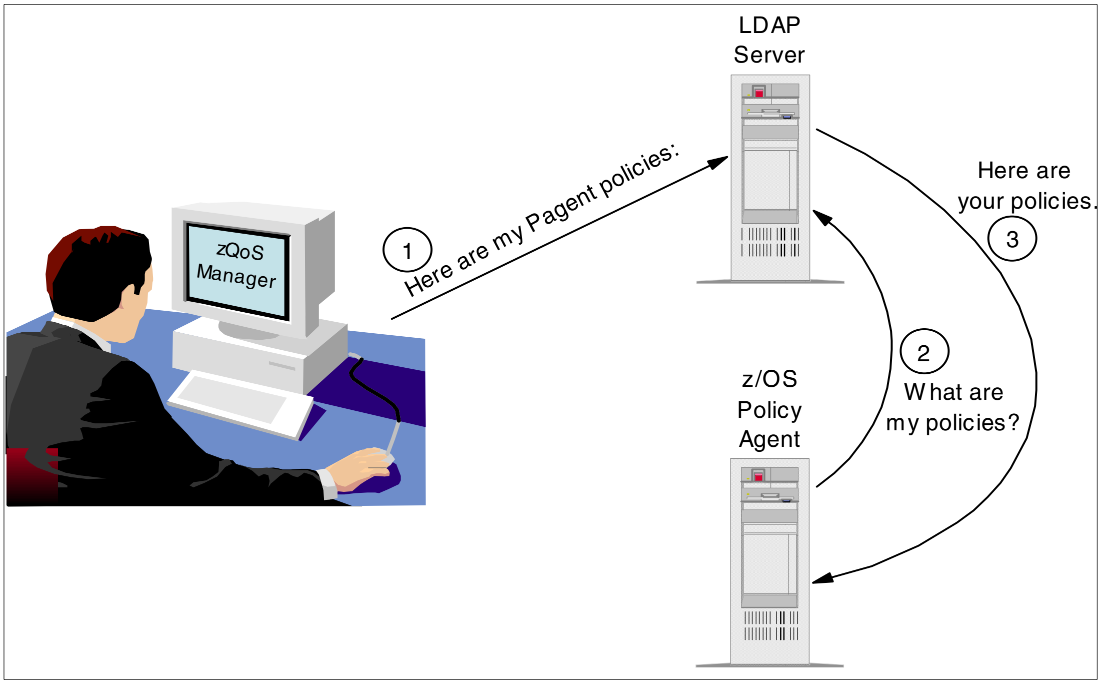


Figure 6-1 Policy flow

zQoS Manager is a tool for network administrators. Therefore, before you begin you should:

- Read the chapter on policy-based networking in *z/OS V1R2 Communications Server: IP Configuration Guide,* SC31-8775.
- Have information about the LDAP Server to be used. For example, the server address and port number, the LDAP protocol version (2 or 3), whether a backup LDAP Server is used, and whether SSL is used.

Be familiar with your particular environment so that you can make decisions on what events are to be detected under what circumstances and what action to take.

6.2 Requirements and support

This section outlines the requirements and support for the QoS GUI.

6.2.1 Requirements

The zQoS Manager requires Java 1.3.1 and Windows 2000 or Linux to run. The Java executable can be obtained at the following URL:

`http://java.sun.com/`

6.2.2 Support - Legal notice

IBM provides this code on an *as is* basis without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability or fitness for a particular purpose.

Support is provided on a “best effort” basis through a newsgroup. Visit the newsgroup ibm.software.commserver.os390.zids-manager on server news.software.ibm.com.

The tested operating systems have been Linux and Windows 2000. Any operating system that can support Java 1.3.1 should be able to run the application.

6.3 Download and installation

The download and installation instructions are written for Windows 2000 and Linux. The information and executables in the following sections are also located at:

`http://www-3.ibm.com/software/network/commserver/downloads/zqosmanager.html`

6.3.1 Windows 2000 steps

The Windows 200 steps are:

1. Download this file to your Windows system: zQoSManager.zip.
2. Use an unzip program to extract zQoSManager.exe.
3. Execute zQoSManager.exe.
4. Go to **Start -> Programs-zQoS Manager**.

6.3.2 Linux steps

The Linux steps are:

1. Download this file to your Linux system: zqosmgr.tar.
2. Untar the file with tar -xvf zqosmgr.tar.
3. Execute ./zqosmgr.

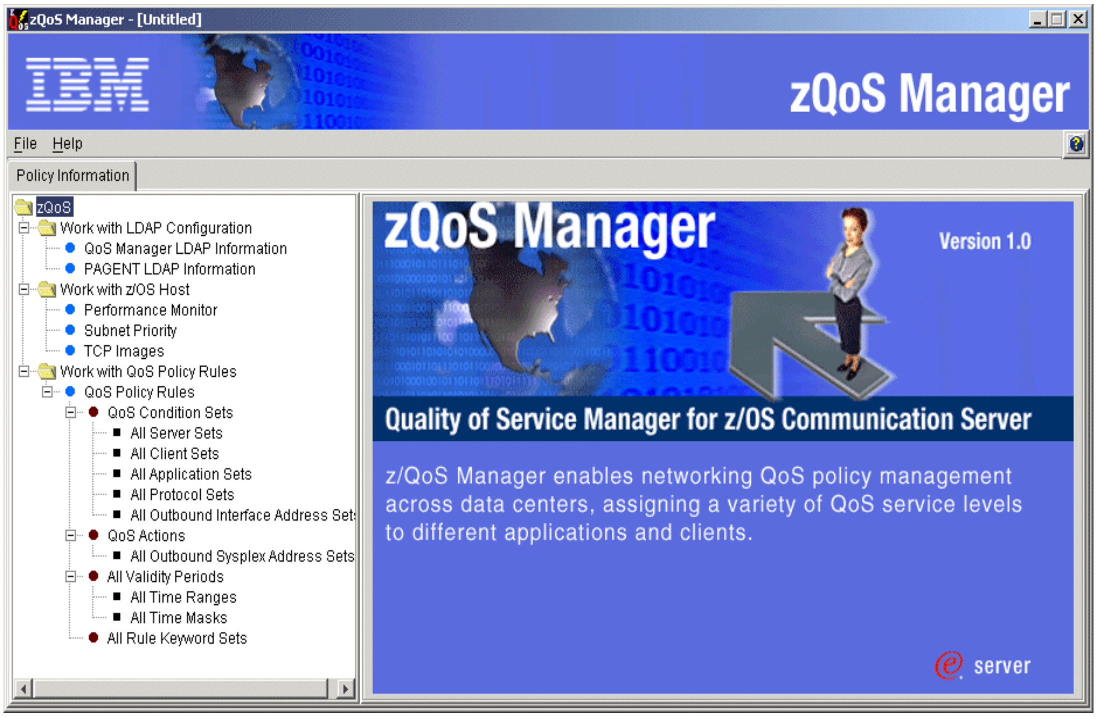
6.4 Using the GUI

This section is intended to help the network administrator manage and understand the Graphical User Interface provided. Each first level directory will be discussed and screen captures provided to assist in the education. The first level directories are:

- 6.4.1, “Work with LDAP configuration” on page 129
- 6.4.2, “Work with z/OS host information” on page 131
- 6.4.3, “Work with QoS policy rules” on page 135

The first window displayed when starting the zQoS Manager is shown in Figure 6-2 on page 129.

> **Note:** zQoS Manager Help is available via the **Help** button. If detailed information is needed for a particular field, place the cursor in the desired field and press the F1 key.



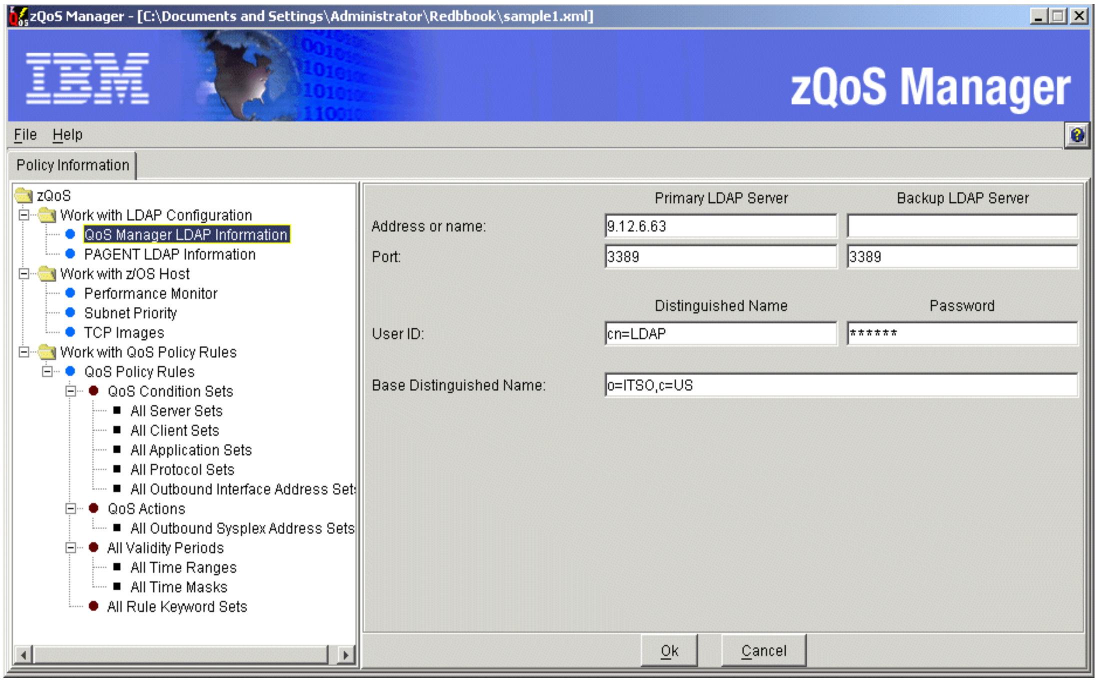
z/QoS Manager enables networking QoS policy management across data centers, assigning a variety of QoS service levels to different applications and clients.

Figure 6-2 zQoS Manager

6.4.1 Work with LDAP configuration

One of the first steps in using zQoS Manger is to configure the LDAP Server settings. This is done from the section **Work with LDAP Information** by selecting **QoS Manager LDAP Information**.

The settings we used are shown in Figure 6-3 on page 130. This is setting up the configuration information that is needed for communication between the zQoS Manager and the LDAP Server.



	Primary LDAP Server	Backup LDAP Server
Address or name:	9.12.6.63	
Port:	3389	3389
	Distinguished Name	Password
User ID:	cn=LDAP	******
Base Distinguished Name:	o=ITSO,c=US	

Figure 6-3 zQoS LDAP configuration information

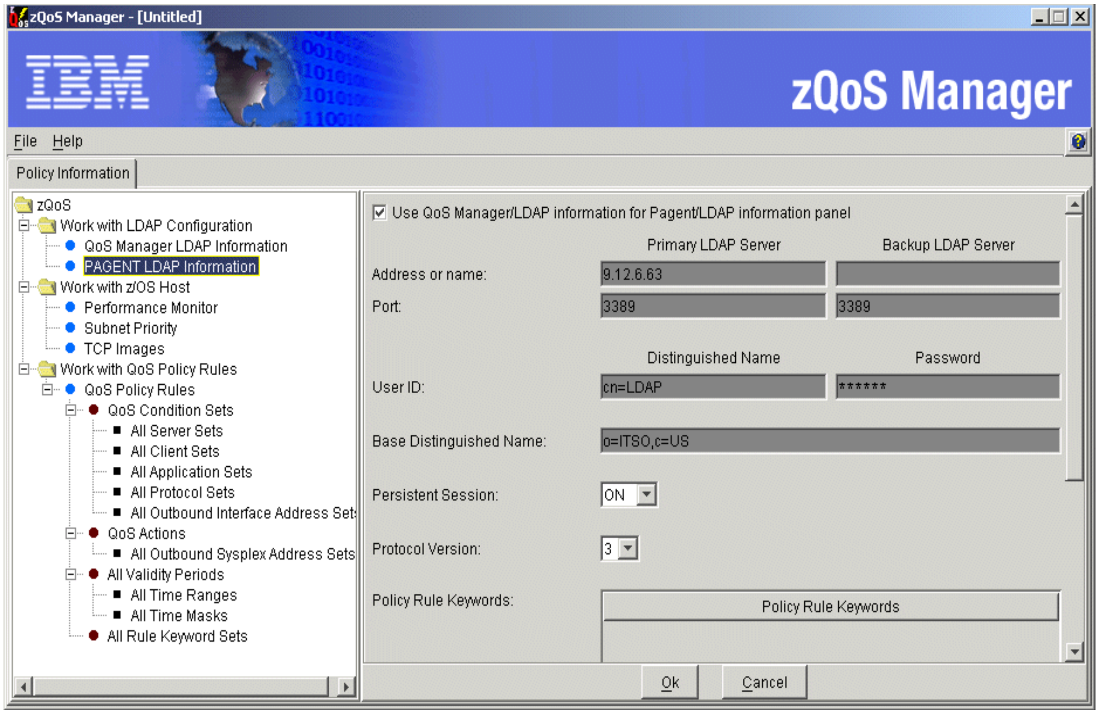
> **Note:** There is not an option for SSL. The connection between the zQoS Manager and the LDAP Server is a non-secure connection.

zQoS Manager to LDAP Server Communication

Verify that you can communicate with the LDAP Server by clicking **File -> Send to LDAP**. This is considered successful if you receive the Creating LDIF icon with the corresponding text `Updating LDAP. Please wait` following for a few seconds, and then are returned control with the icon disappearing. Keep in mind that we not sending any new policy information to the LDAP Server, we basically sent the default IBM-provided policy information for connectivity verification. This same step must be repeated after the policies have been coded and saved to a file, **File -> Save** or **File -> Save As**.

PAGENT LDAP information

Next we will configure the PAGENT configuration information by selecting **PAGENT LDAP Information**. See Figure 6-4 on page 131. In this panel if the Use QoS Manager/LDAP information for PAGENT/LDAP information panel box it ticked then the zQoS Manager automatically reuses the information that you specified in the ‘QoS Manager LDAP information’ panel. To change this information simply remove the tick from the box. You also enter other PAGENT configuration information in this panel, such as PAGENT to maintain a persistent session with the LDAP Server, is the PAGENT/LDAP Server connection to use SSL. Also what protocol version is PAGENT to use to communicate with the LDAP Server (is the LDAP Server a type 2 or type 3).



☑ Use QoS Manager/LDAP information for Pagent/LDAP information panel

	Primary LDAP Server	Backup LDAP Server
Address or name:	9.12.6.63	
Port:	3389	3389
	Distinguished Name	Password
User ID:	cn=LDAP	******

Base Distinguished Name: o=ITSO,c=US

Persistent Session: ON

Protocol Version: 3

Policy Rule Keywords:

Figure 6-4 PAGENT LDAP configuration information

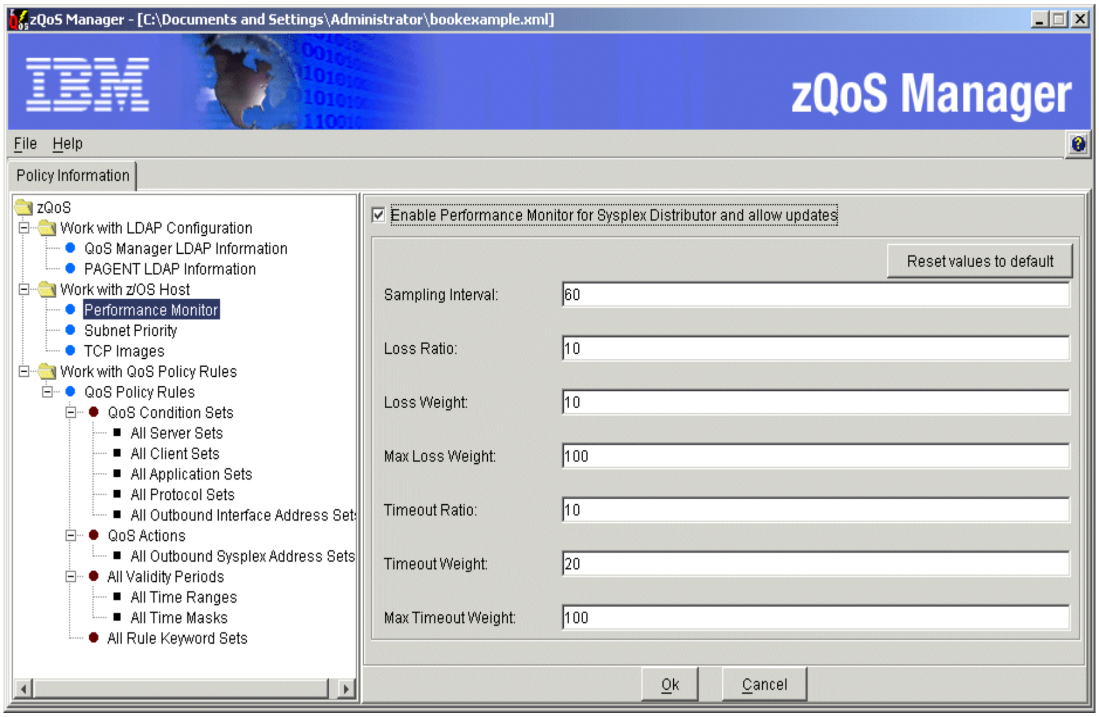
6.4.2 Work with z/OS host information

This section provides additional information that will be included in PAGENT's configuration file on the z/OS host. The information that you provide will depend upon your policies. The factors to consider are:

- Should policies be applied to Sysplex Distributor?
- Are you using OSA Express cards in QDIO mode?
- Are you running with multiple TCP/IP stacks on this LPAR? And if so, to which instances of TCP/IP should your policies be applied?

Performance monitor

If you tick Enable Performance Monitor for Sysplex Distributor and allow updates, this option allows you to assign weight fractions to the monitored performance data and send them to the Sysplex Distributor distributing stack as the monitored data crosses the defined threshold.



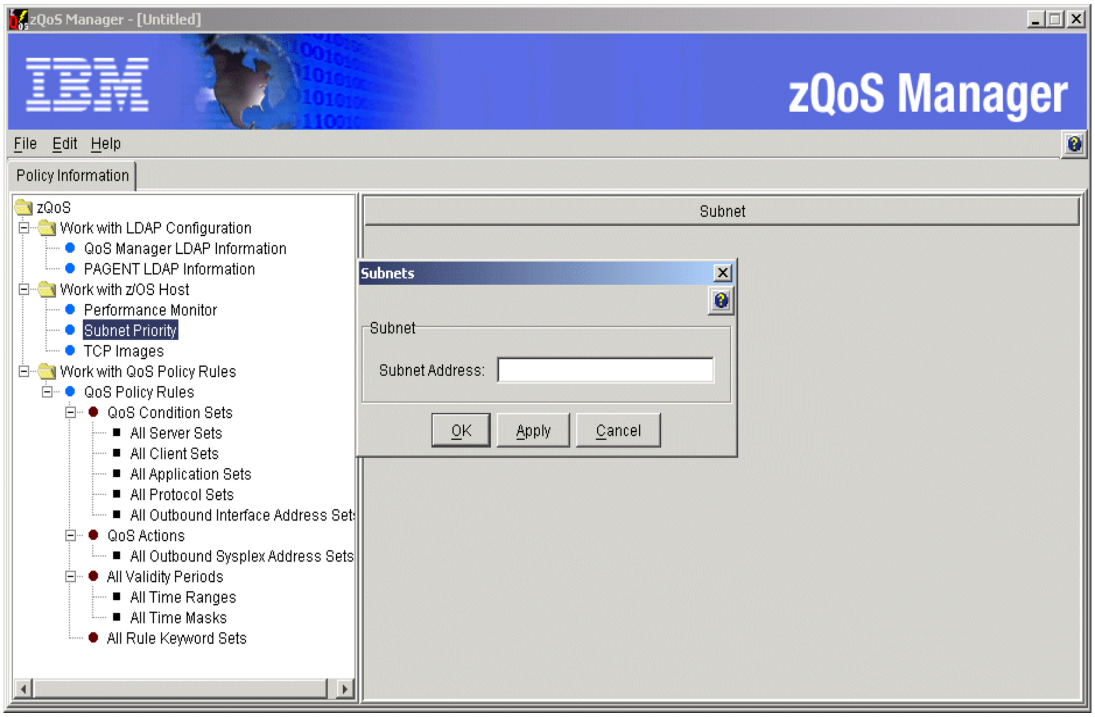
☑ Enable Performance Monitor for Sysplex Distributor and allow updates

Sampling Interval:	60
Loss Ratio:	10
Loss Weight:	10
Max Loss Weight:	100
Timeout Ratio:	10
Timeout Weight:	20
Max Timeout Weight:	100

Figure 6-5 Performance monitor for Sysplex Distributor

Subnet priority

This option is used to define mappings of IP ToS/DSCP for each OSA Express card configured in QDIO mode for outbound user interface priorities and to VLAN user priorities. To add a subnet select **Edit -> Add a Subnet**.

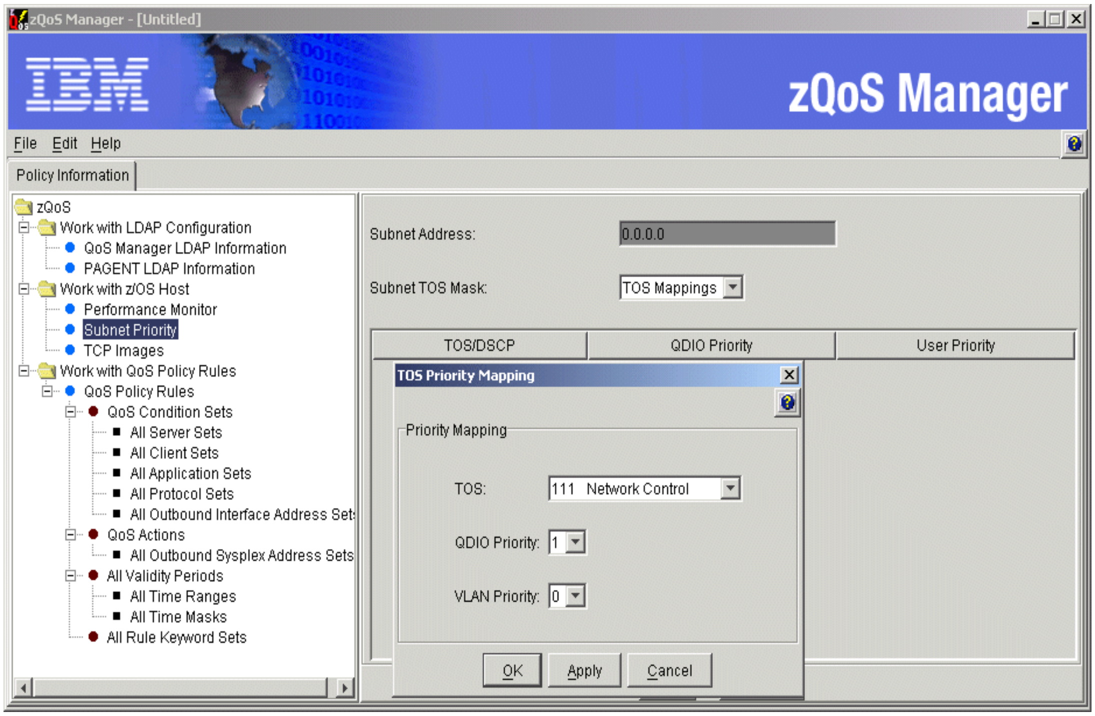


Subnet

Subnet Address:

Figure 6-6 Adding a subnet

Once the subnet has been added, select **Edit -> Work with subnet**, then **Edit -> Add TOS/DSCP Priority Mappings**, and then add the TOS/DSCP priority mappings as shown in Figure 6-7.



Subnet Address: 0.0.0.0

Subnet TOS Mask: TOS Mappings

Priority Mapping

TOS: 111 Network Control

QDIO Priority: 1

VLAN Priority: 0

Figure 6-7 Adding TOS priority mappings

TCP images

This panel selection allows you to specify which TCP/IP images you want the policies defined in this PAGENT configuration to be applied to. This is useful if you are running multiple instances of TCP/IP on a single MVS image or if you will use a single PAGENT Configuration data set/file in a multisystem sysplex.

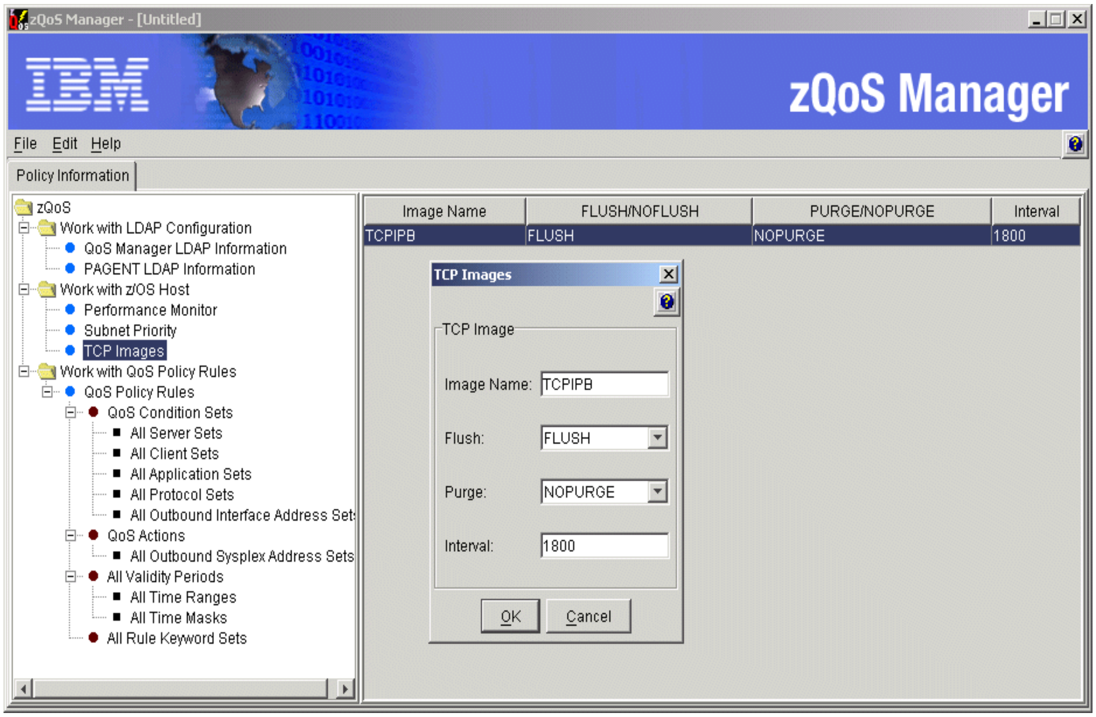


Image Name	FLUSH/NOFLUSH	PURGE/NOPURGE	Interval
TCPIPB	FLUSH	NOPURGE	1800

TCP Images

TCP Image

Image Name: TCPIPB

Flush: FLUSH

Purge: NOPURGE

Interval: 1800

Figure 6-8 Adding TCP images

At this point we are ready to generate the PAGENT configuration file.

PAGENT configuration file

The PAGENT configuration information must be saved to a text file. After providing the necessary information, which we did above. The file is saved by doing a **File -> Save As** and selecting the .conf format, which represents the PAGENT LDAP configuration file. An example of the text output is displayed in Figure 6-9 on page 135.

```
ReadFromDirectory
{
     LDAP_Server                  9.12.6.63
     LDAP_Port                    3389
     LDAP_DistinguishedName       cn=LDAP
     LDAP_Password                secret
     LDAP_SessionPersistent       Yes
     LDAP_ProtocolVersion         3
     LDAP_SchemaVersion           3
     SearchPolicyBaseDN           o=ITSO,c=US
}

PolicyPerfMonitorForSDR Enable
{
     SamplingInterval 60
     LossRatioAndWeightFr 10 10
     LossMaxWeightFr 100
     TimeoutRatioAndWeightFr 10 20
     TimeoutMaxWeightFr 100
}

SetSubnetPrioTosMask
{
     SubnetAddr 0.0.0.0
     SubnetTosMask 11100000
     PriorityTosMapping 1 11100000 0
     PriorityTosMapping 1 11000000 0
     PriorityTosMapping 1 10100000 0
     PriorityTosMapping 1 10000000 0
     PriorityTosMapping 2 01100000 0
     PriorityTosMapping 3 01000000 0
     PriorityTosMapping 4 00100000 0
     PriorityTosMapping 4 00000000 0
}

TcpImage TCPIPB FLUSH NOPURGE 1800
```

Figure 6-9 PAGENT configuration file

This file is the PAGENT configuration file as described in 4.2, “Coding policy definitions in a configuration file” on page 77. This information must be manually transferred (that is, FTPed, cut and pasted, or retyped) to the PAGENT configuration file, located on the z/OS V1R2.0 system. Typically the /etc/pagent.conf file is used when the PAGENT application is started.

6.4.3 Work with QoS policy rules

This section is where you specify the QoS policy rules. This is the most critical task and typically will be done iteratively until the final policy rules are accepted.

- You are required to establish one Condition Set and one action set in at least one policy rule.
- You may optionally specify that rules apply only during validity periods.
- You may optionally associate rules with keywords to speed up their retrieval from LDAP.

Use this section to specify QoS policy rules, which can include Condition Sets, actions, policy keyword sets, or validity periods. However, only one policy rule and associated actions can be applied to a particular unit of network traffic.

When you have finished specifying QoS policy rules, select **File -> Send to LDAP** to store the policy information in the LDAP Server. Also use the **File -> Save** as tab and select the **PAGENT LDAP Configuration files (.conf)** pop-up to save the pagent.conf file used by PAGENT.

At anytime when setting up your policies with the zQoS Manager, you may save intermediate policy information in an XML file on the workstation running zQoS Manager, by selecting **File -> Save As** and selecting the file type as XML.

> **Note:** Only XML files can be read back in the zQoS Manager and edited. LDIF and CONF files are generated and are not reusable by the zQoS Manager.

QoS policy rules

This section gives you the ability to actually create a policy rule by linking together policy conditions with policy actions and validity periods (time periods when the policy condition will be active). You also have the option of identifying whether this particular rule will be used by the Sysplex Distributor for load distribution. However, before you get to this stage you must create the QoS Condition Sets, the QoS actions, and all validity ranges.

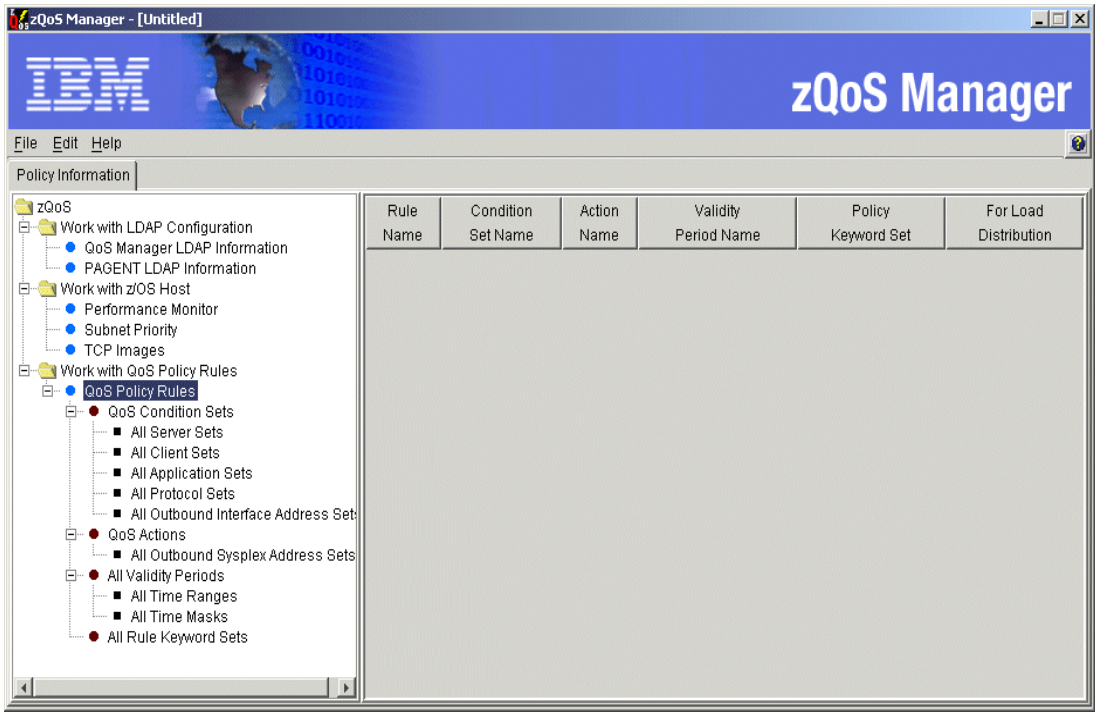


Figure 6-10 QoS policy rules

QoS Condition Sets

Like the QoS policy rules, the QoS Condition Sets link together sets of information. In this case these are all actual conditions that you will want PAGENT to check for. As you can see in Figure 6-11 on page 137, you identify the server, client, application, and protocol and outbound interface sets to a QoS Condition Set name. When creating a QoS Condition Set you do not need to include an entry for all the different sets, just the ones that you want to be included in this rule (minimum of 1).

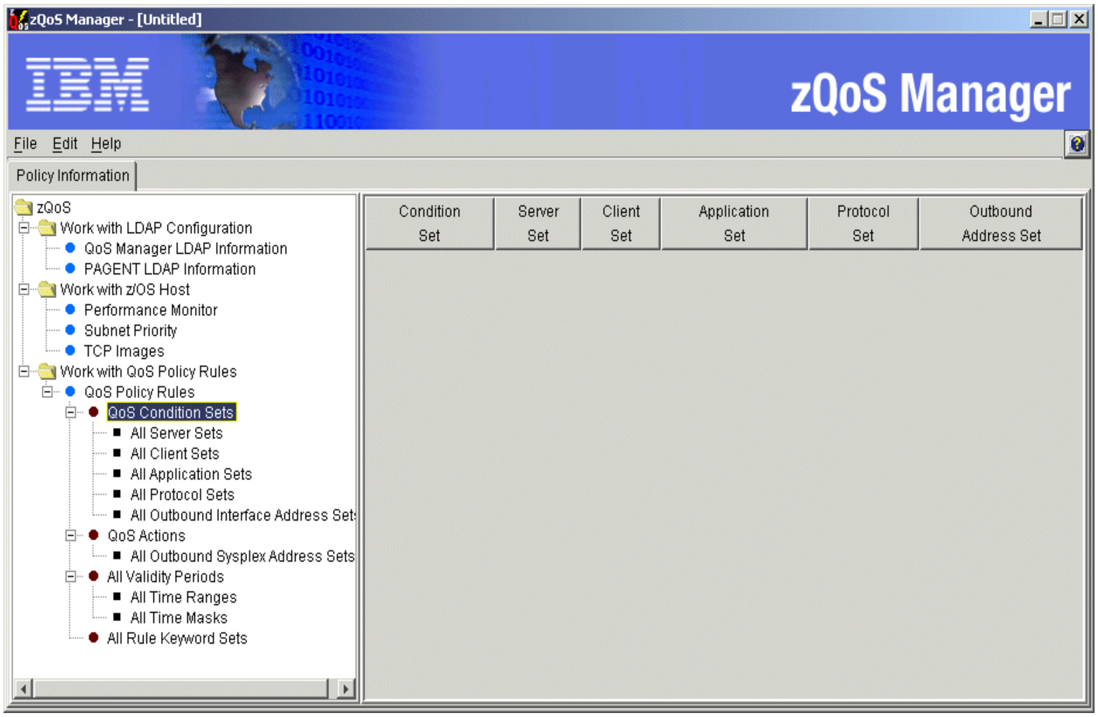
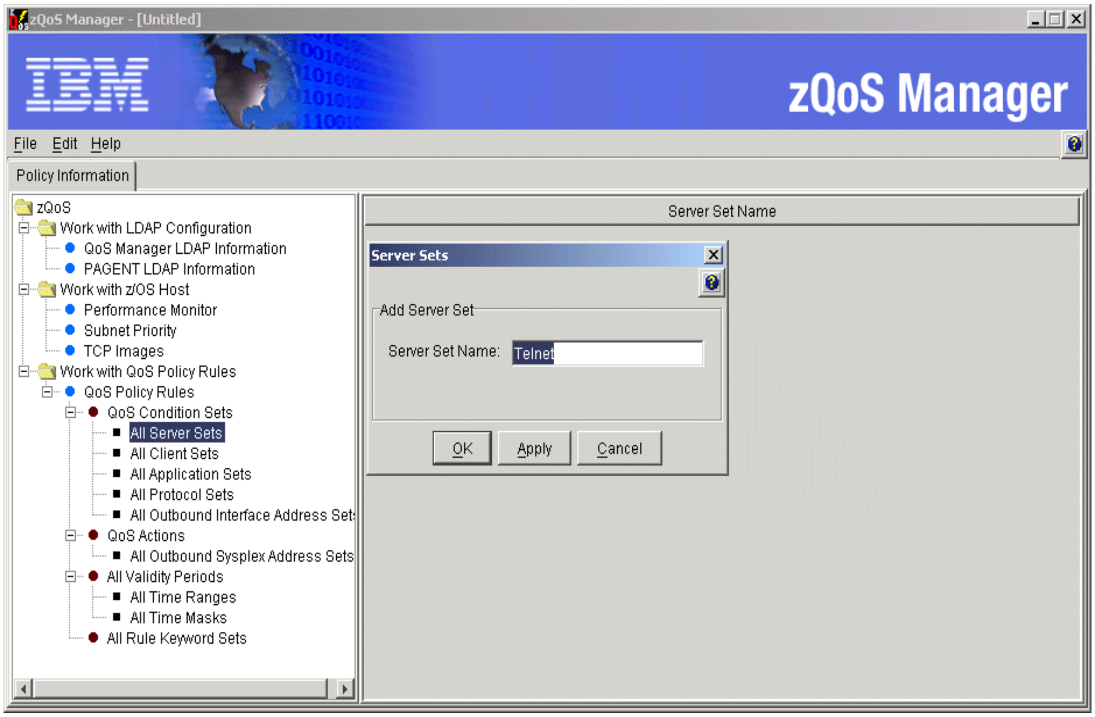


Figure 6-11   QoS Condition Sets

Creating a QoS Condition Set

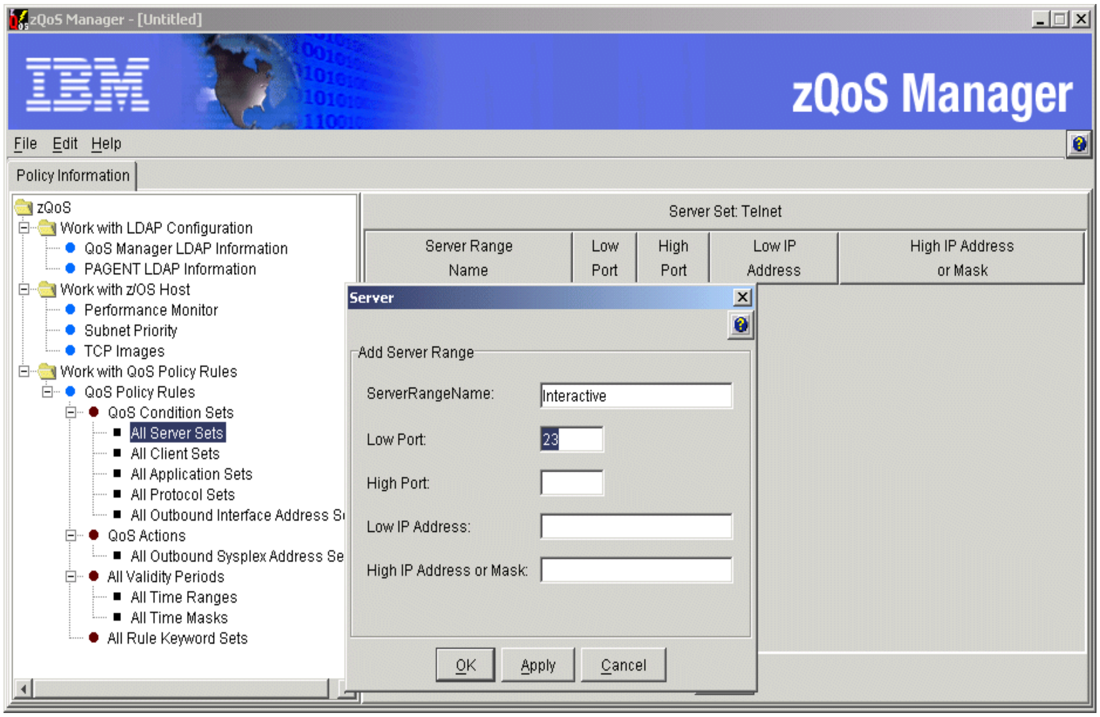
To add a server set, select **Edit -> Add Server set** and then enter a name for this server set. In this example we used the name Telnet, as shown in Figure 6-12.



Server Set Name: Telnet

Figure 6-12   Add server set

Once the Server set has been added, highlight the server set you want to work with. Next, select **Edit -> Work with Server set** to bring you to, in our case, the server set Telnet panel. Here you select **Edit -> Add Server range**. The screen will appear as shown in Figure 6-13.



ServerRangeName: Interactive

Low Port: 23

High Port:

Low IP Address:

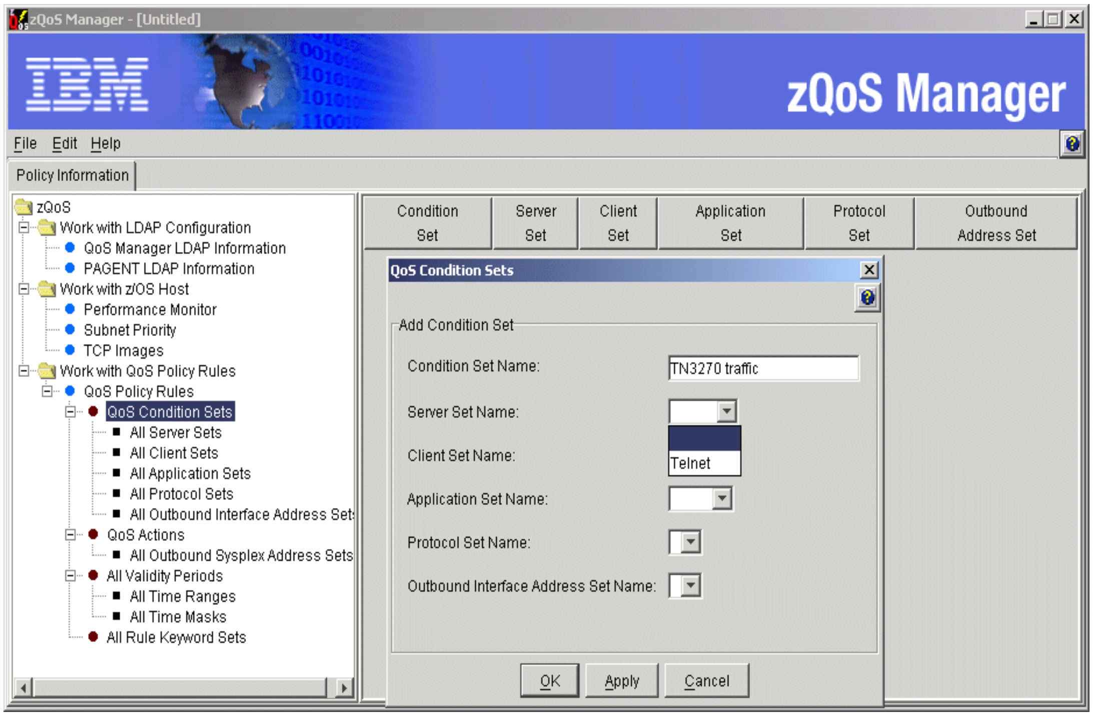
High IP Address or Mask:

Figure 6-13 Add server range

We will not walk you through the configuration of all the possible QoS Condition Set entries, as once you start using the zQoS Manager, it will become clear as to how to add them.

To link the Condition Sets entries into a QoS Condition Set select **Edit -> Add New Condition Set**. Now specify the QoS Condition Set name you want to use, and by clicking the down arrow select the Condition Set entry you want to be included in this QoS Condition Set. In our example, Figure 6-14 on page 139, we only have the Application Server Set, Telnet, which may be included.

Note: Please be aware that these Condition Sets are reusable and as such may be included in multiple QoS Condition Sets, as is the QoS Condition Set itself, which may be included in multiple QoS policy rules.



Condition Set Name: TN3270 traffic

Server Set Name:

Client Set Name: Telnet

Application Set Name:

Protocol Set Name:

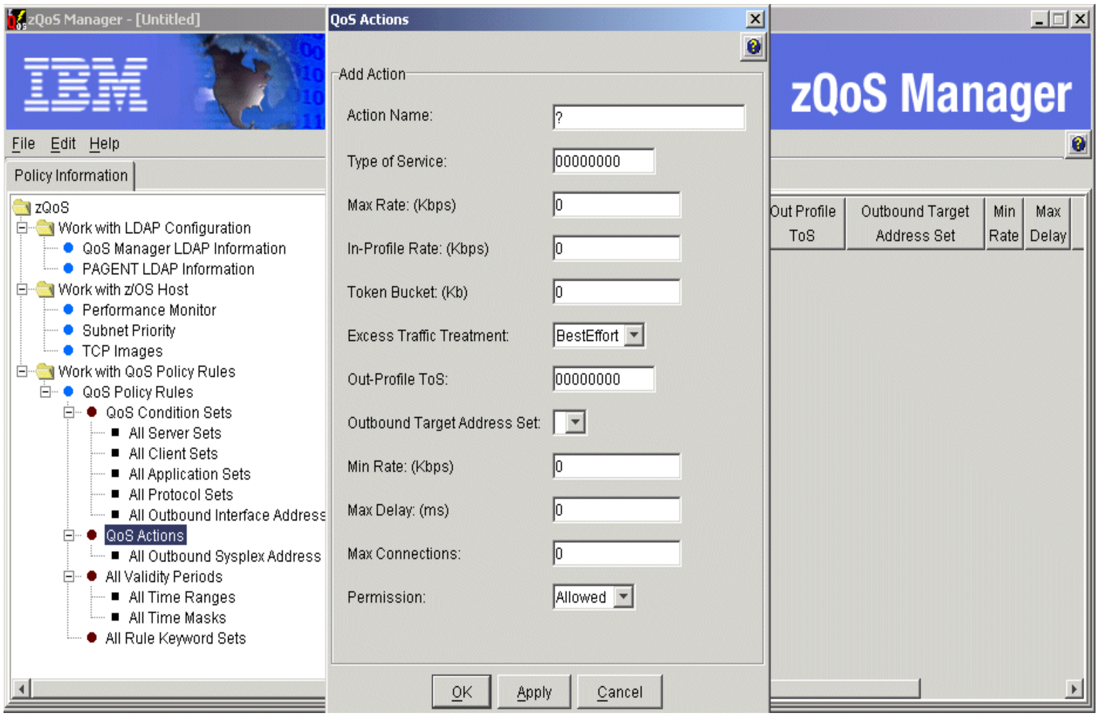
Outbound Interface Address Set Name:

Figure 6-14   Adding a QoS Condition Set

QoS actions

Now that you have created your policy rule, the next step is to add policy actions. This is done through the QoS Actions panels. Here we define what should happen to a packet if it matches a particular policy rule.

To add an action, from the QoS Actions panel select **Edit -> Add New Action**. The menu appears as shown in Figure 6-15 on page 140.

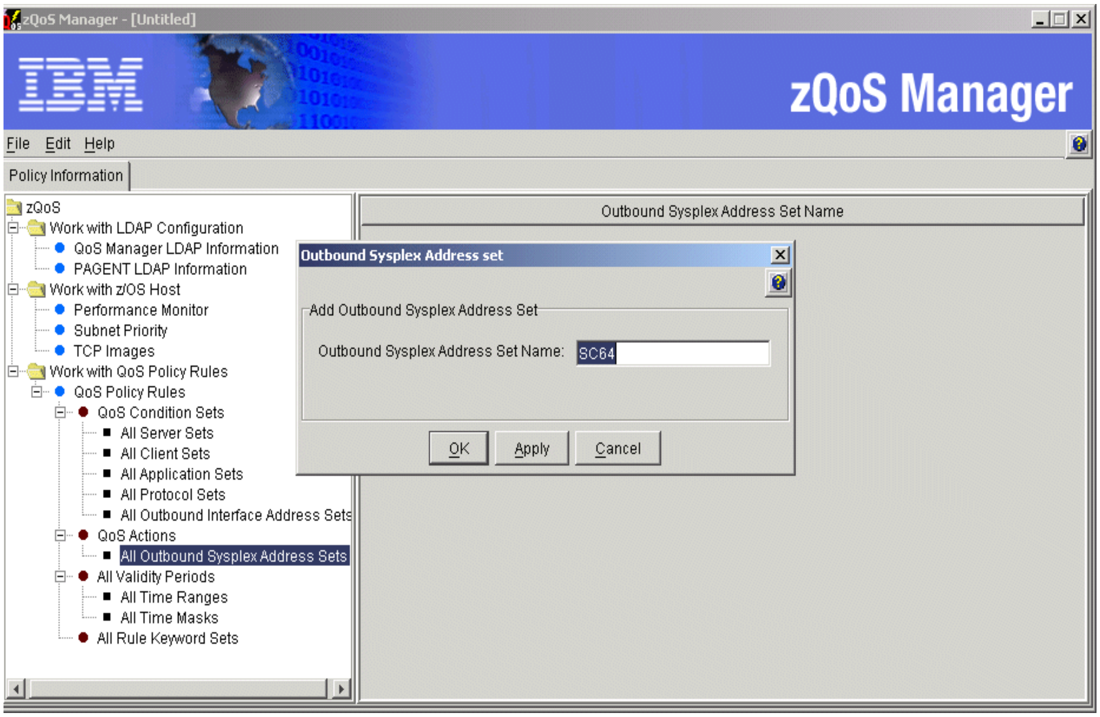


Add Action

Action Name:	?
Type of Service:	00000000
Max Rate: (Kbps)	0
In-Profile Rate: (Kbps)	0
Token Bucket: (Kb)	0
Excess Traffic Treatment:	BestEffort
Out-Profile ToS:	00000000
Outbound Target Address Set:	
Min Rate: (Kbps)	0
Max Delay: (ms)	0
Max Connections:	0
Permission:	Allowed

Figure 6-15 Adding a QoS action

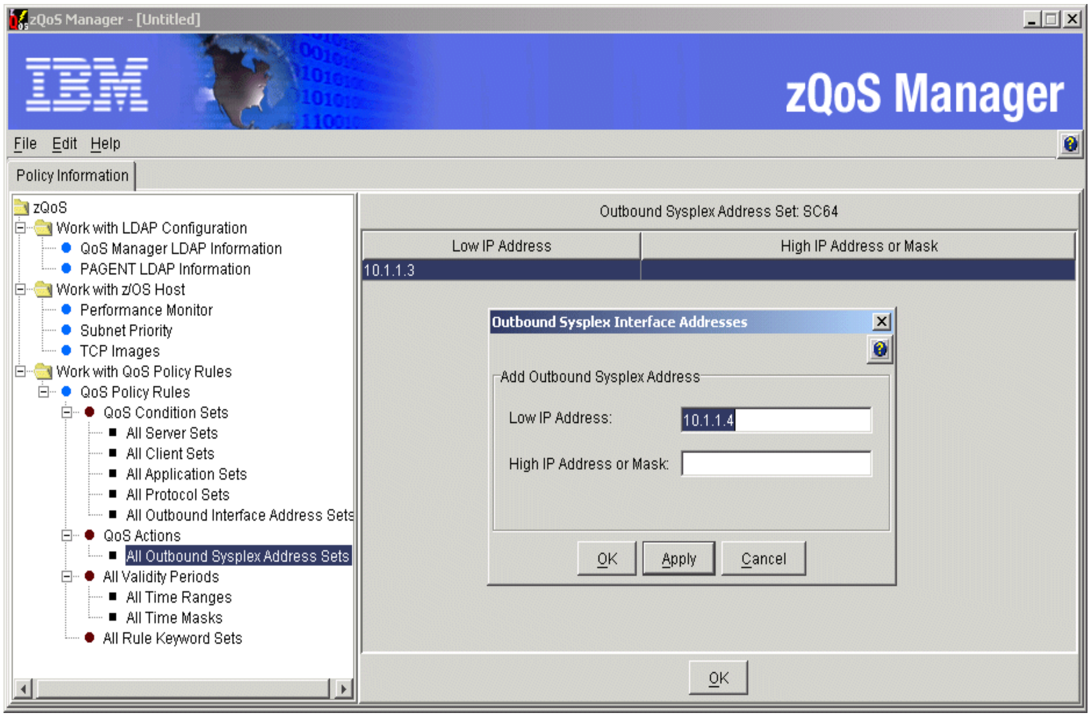
Here you specify the name of the QoS action and other actions that you would require be applied to a packet. If you are using Sysplex Distributor, and this action is for one of the connections that is being distributed, you have the ability to limit which of the target stack, defined on the VIPADISTRIBUTE statement, this connection may be distributed to. To do this, from All Outbound Interface Address Sets, select **Edit -> Add Outbound Sysplex Address Set**, and enter a name for this interface set. Although you may use any names, for convenience we at the ITSO used the name of our Sysplex Distributing stack, SC64. This is seen in Figure 6-16 on page 141.



Outbound Sysplex Address Set Name: SC64

Figure 6-16 Add in a sysplex interface name

Once you have defined the Outbound Sysplex Address Set names, highlight one of the names and select **Edit - >Work with Outbound Sysplex Address Set**, then select **Edit -> Add Outbound Sysplex Address**, as shown in Figure 6-17 on page 142. The IP addresses that you code here are the XCF interface addresses of the target TCP/IP stacks, as configured on the VIPADISTRIBUTE statement on the distributing TCP/IP stack. You would only code the XCF interfaces of the systems that you want included as possible destinations for the packets that matched the policy rule that this policy action is for.



Outbound Sysplex Address Set: SC64

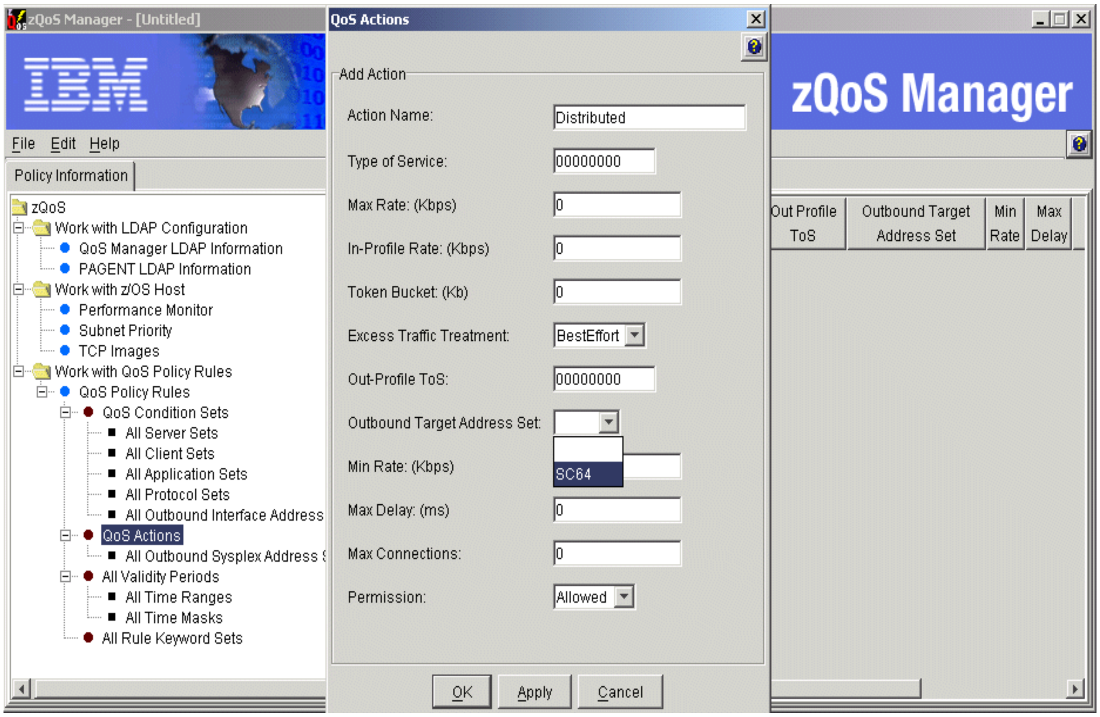
Low IP Address	High IP Address or Mask
10.1.1.3	

Low IP Address: 10.1.1.4

High IP Address or Mask:

Figure 6-17 Add outbound sysplex address

Now that the XCF interfaces you want to use are defined in the Outbound Sysplex Address Set name, you can go back to the QoS Action panel and add the Outbound Target Address Set to the QoS Action Name entry, as seen in Figure 6-18.



Action Name: Distributed

Type of Service: 00000000

Max Rate: (Kbps) 0

In-Profile Rate: (Kbps) 0

Token Bucket: (Kb) 0

Excess Traffic Treatment: BestEffort

Out-Profile ToS: 00000000

Outbound Target Address Set: SC64

Min Rate: (Kbps)

Max Delay: (ms) 0

Max Connections: 0

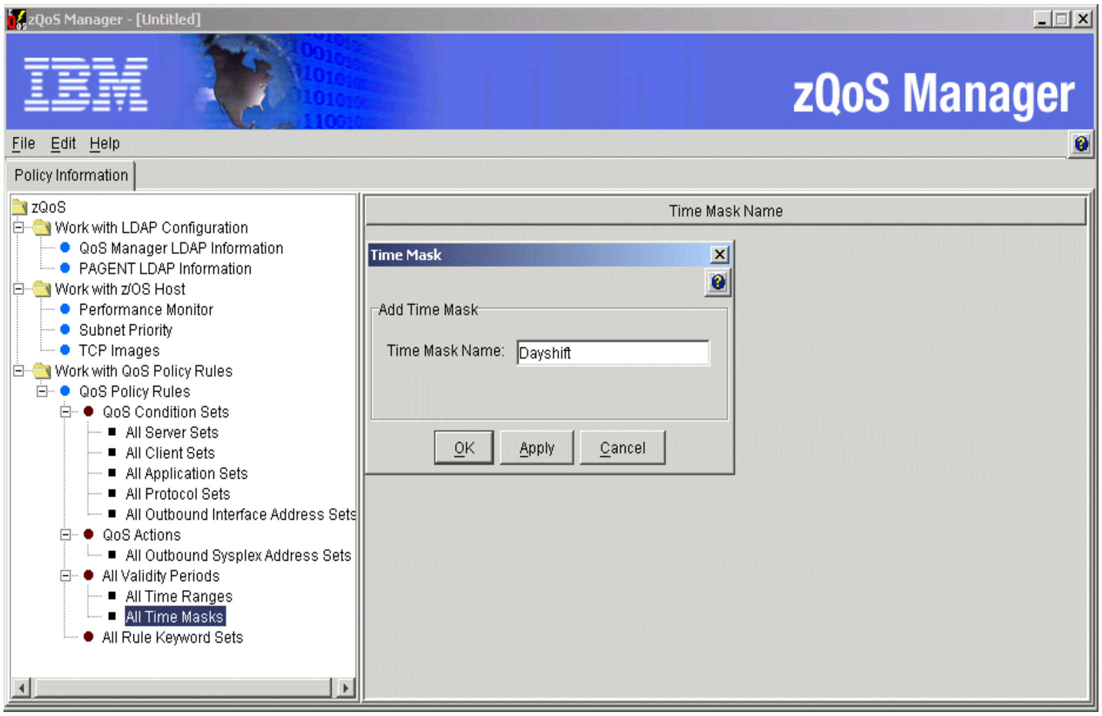
Permission: Allowed

Figure 6-18 Adding an outbound interface address to a policy action

All validity periods

Now that the policy rules and policy actions have been completed, you will need to identify any limitations as to when the policies will be active. If you want your policies to always be active (the default), then you do not need to specify anything in this section.

For this example we want a policy to only be active between 08.00 and 18.00 on week days. You would highlight All Time Masks then select **Edit -> Add Time Mask**. Enter a time mask name, in our case Dayshift, as shown in Figure 6-19.

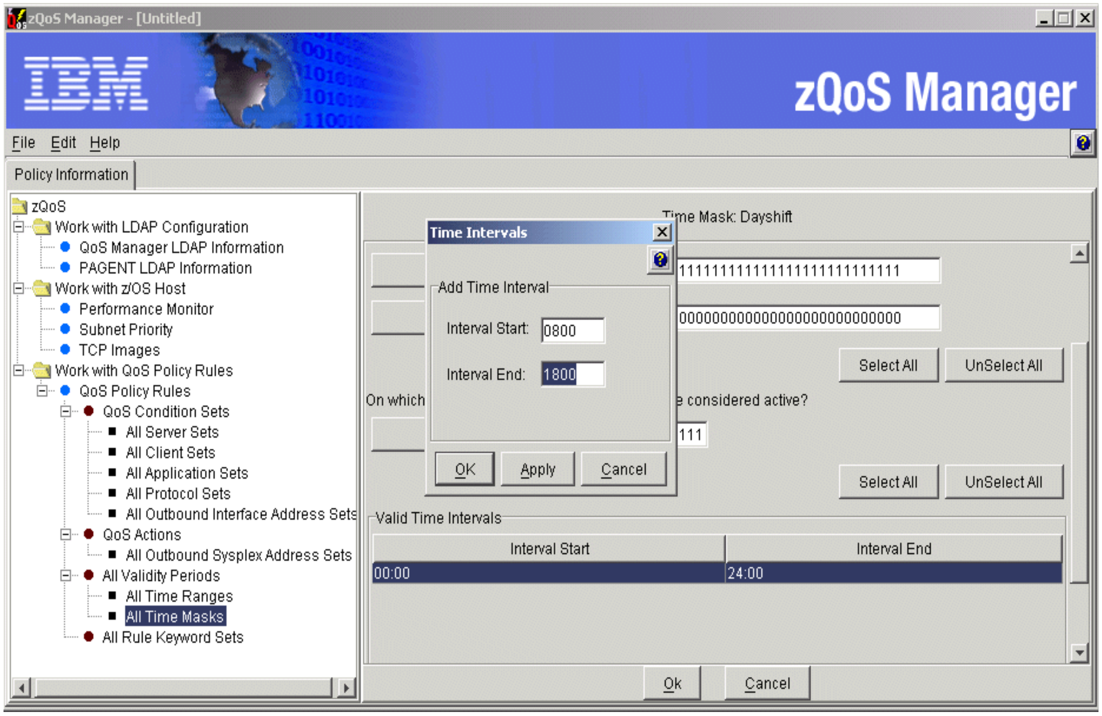


Time Mask Name: Dayshift

Figure 6-19 Adding a time mask name

Highlight the time mask name that you want to work with, then **Edit -> Work with Time Mask**. A panel that comes up and provides you with possible day/month/TOD options for coding the time mask.

Our policy is to be active between 08.00 and 18.00. So select **Edit -> Add Time Interval** and enter an interval start of 0800 and an interval end of 1800, as shown in Figure 6-20 on page 144. You are not limited to coding just a single time interval and may enable and disable a policy rule many times during a day.



Time Mask: Dayshift

11111111111111111111111111

0000000000000000000000000

On which ... e considered active?

111

Add Time Interval

Interval Start: 0800

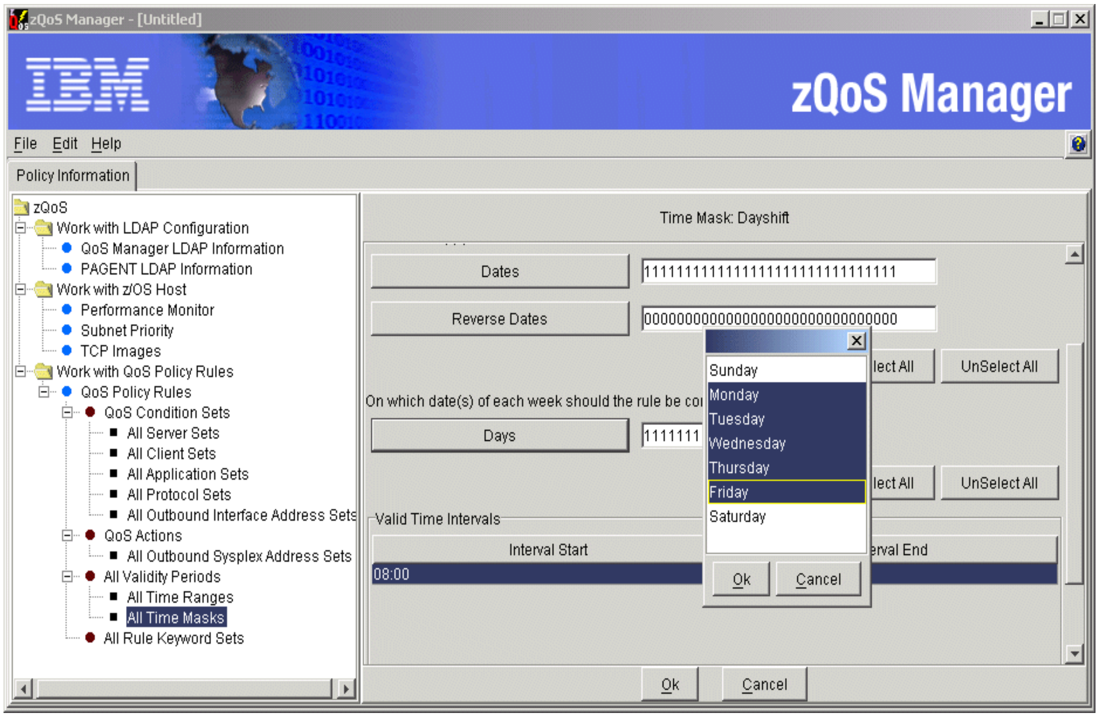
Interval End: 1800

Valid Time Intervals

Interval Start	Interval End
00:00	24:00

Figure 6-20 Adding a time interval

The policy is only to be valid on weekdays. This is set in the Days field. There are a number of ways to update this field; the simplest is to overwrite the existing mask with the mask of 0111110 (bits are Sunday through Saturday). You can also select the **Days** button and highlight the days you want active (hold down the shift or control key for multiple days), as shown in Figure 6-21 on page 145.



Time Mask: Dayshift

Dates 1111111111111111111111111111111

Reverse Dates 0000000000000000000000000000000

On which date(s) of each week should the rule be co

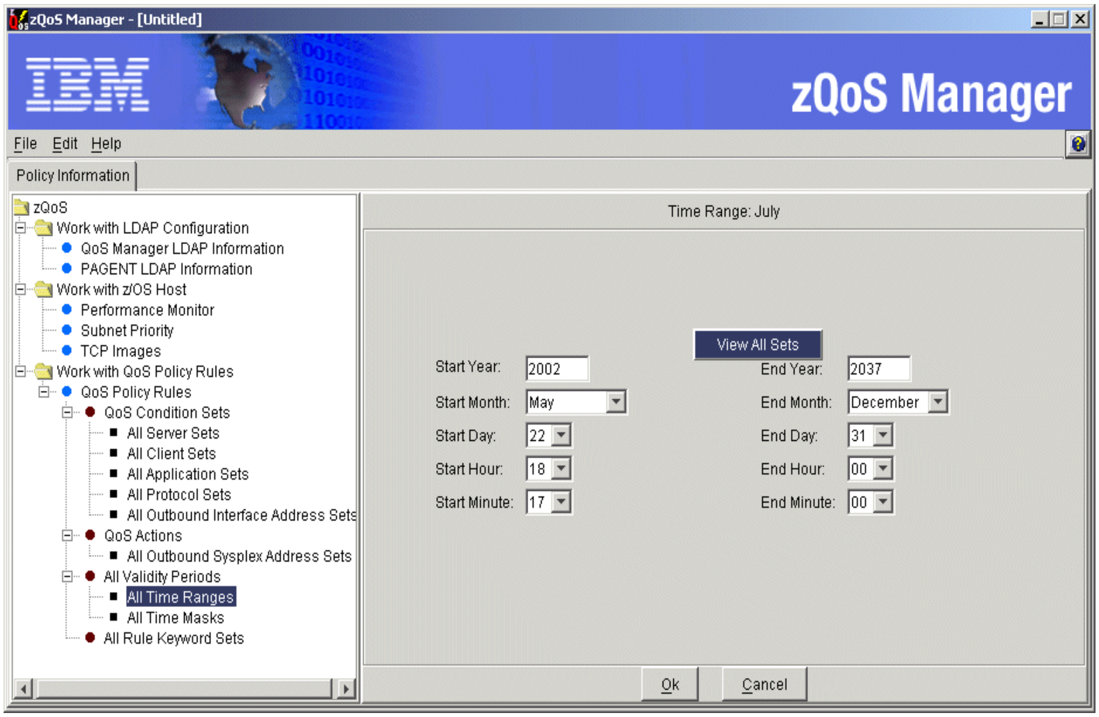
Days 1111111

Valid Time Intervals

Interval Start	erval End
08:00	

Figure 6-21 Selecting weekdays

If you only want a particular policy to be active between specific dates, highlight All Time Ranges in the left-hand panel, select **Edit -> Add Time Range** and enter the name you want to use for this date range. Highlight the time range name in the right-hand panel, select **Edit -> Work with Time Range**, and enter a name for the time range and the starting/ending date/time when you want a policy to be active, as shown in Figure 6-22 on page 146.



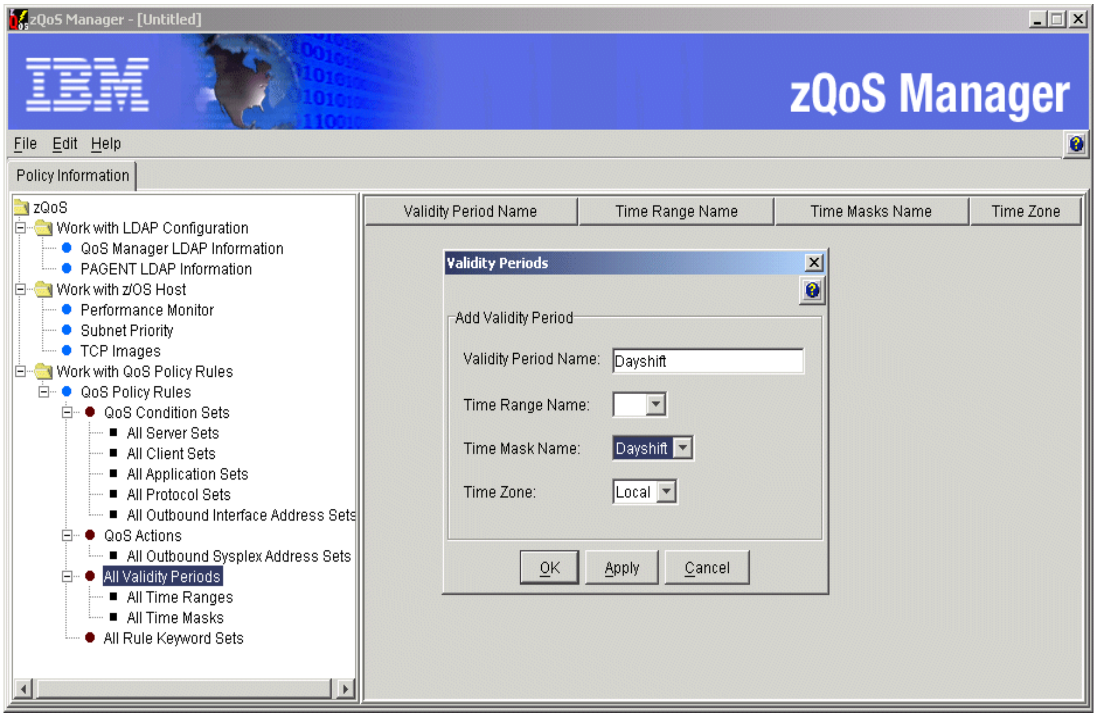
Time Range: July

Start Year: 2002
Start Month: May
Start Day: 22
Start Hour: 18
Start Minute: 17

End Year: 2037
End Month: December
End Day: 31
End Hour: 00
End Minute: 00

Figure 6-22 Adding a time range

Now that you have a time mask and/or a time range, you can add a validity period entry. In the left-hand panel highlight All Validity Periods, select **Edit -> Add New Validity Period**, and enter the name you want to assign to this validity period, the time range, and/or the time mask, as shown in Figure 6-23.



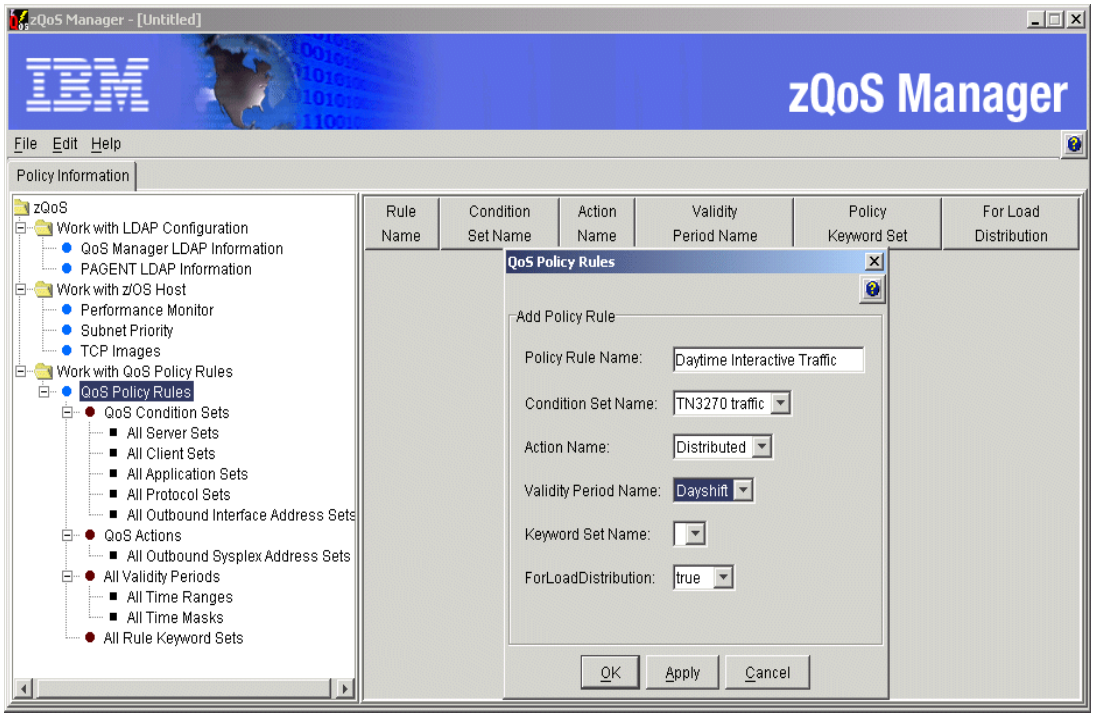
Add Validity Period

Validity Period Name: Dayshift
Time Range Name:
Time Mask Name: Dayshift
Time Zone: Local

Figure 6-23 Specifying a validity period

Creating a QoS policy rule

Now that we have all the individual components and a policy rule name, a policy action name, and a Policy Time Period name, you can link them all together into a policy rule. To do this you will need to highlight QoS Policies Rules in the left-hand panel, as shown in Figure 6-10 on page 136, select **Edit -> Add Policy Rule**, and specify whether you want this new rule to appear above or below the currently highlighted policy rule in the list. You are now presented with a panel where you enter the new policy rule name and select a Condition Set Name, an Action Name, a Validity Period Name, and a Keyword name (if you have defined one), from the drop-down list of entries that you went through and defined earlier. You also identify, by selecting **ForLoadDistribution TRUE**, whether the policy is to be Sysplex Distributor load distribution.



Policy Rule Name: Daytime Interactive Traffic

Condition Set Name: TN3270 traffic

Action Name: Distributed

Validity Period Name: Dayshift

Keyword Set Name:

ForLoadDistribution: true

Figure 6-24 Defining a policy rule

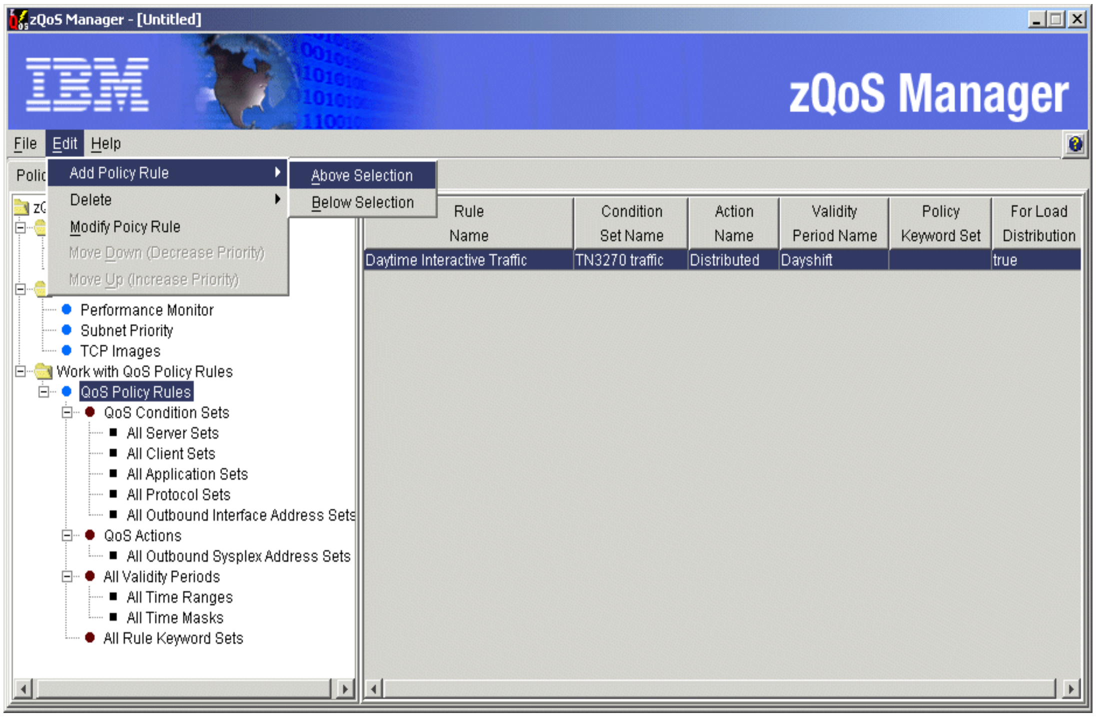
When you have created the policy rules you desire, and assuming you have connectivity to the LDAP Server, see “zQoS Manager to LDAP Server Communication” on page 130. You can send the new policy rules to the LDAP Server by simply selecting **File -> Send to LDAP**.

6.5 Policy priorities

Policies consist of several related objects. The main object is the policy rule. A policy rule object refers to one or more policy condition, policy action, or policy validity period objects. Validity periods determine when each policy rule is active. Active policy objects are analogous to an IF statement in a program. For example:

```
IF condition THEN action
```

In other words, when the set of conditions referred to by a policy rule are TRUE, then the policy actions associated with the policy rule are executed. Only one policy rule and associated action can be applied to a particular packet. The prioritization of the policy can be seen when you add a policy and receive the **Above Section/Below Section** option, as shown in Figure 6-25.



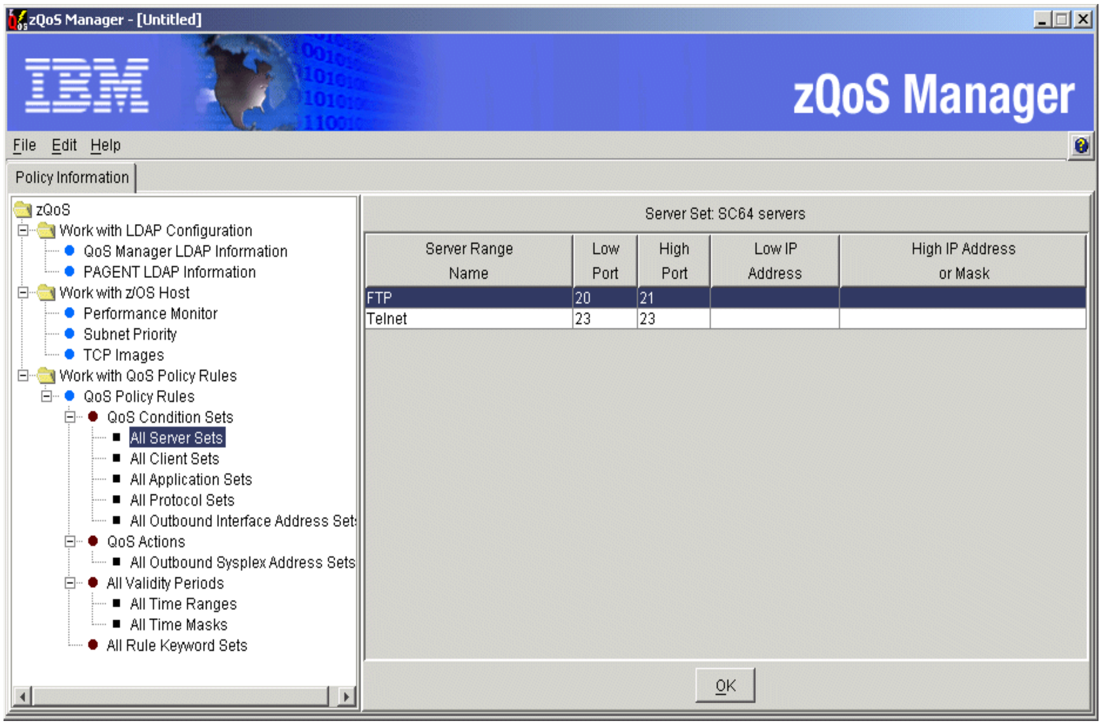
Rule Name	Condition Set Name	Action Name	Validity Period Name	Policy Keyword Set	For Load Distribution
Daytime Interactive Traffic	TN3270 traffic	Distributed	Dayshift		true

Figure 6-25 Policy prioritization

The first policy with a true condition will be executed, thus, the prioritization of policies must be evaluated prior to implementation. One can easily prioritize a policy by clicking a policy in the right hand pane and when the user chooses to add a policy, they are prompted with a specification for Above/Below the current policy. In other words, does this policy have a higher or lower priority than the active policy.

6.5.1 Conjunctive Normal Form (CNF) policies

The zQoS Manager only supports the Conjunctive Normal Form, which means an ANDed (different condition levels) set of ORed conditions (same condition level). This ORing of the same level conditions can be seen in Figure 6-26 on page 149. For further information on CNF please refer to see 1.3, “Policy model overview” on page 5.



Server Set: SC64 servers

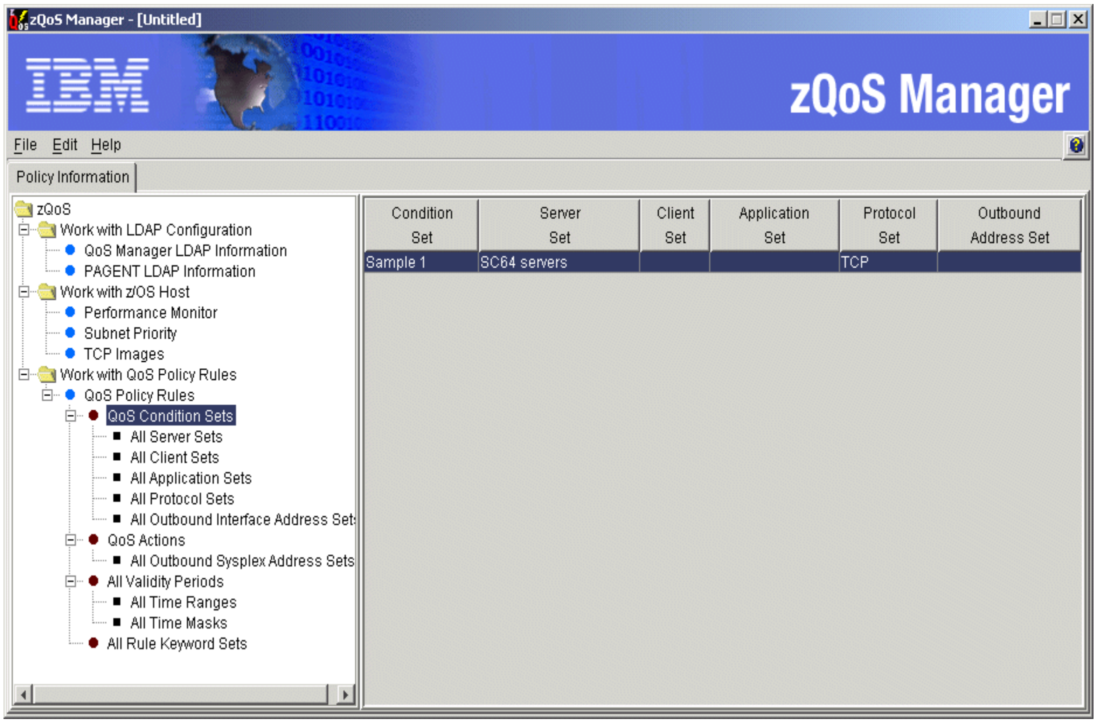
Server Range Name	Low Port	High Port	Low IP Address	High IP Address or Mask
FTP	20	21		
Telnet	23	23		

Figure 6-26 CNF ORed condition

The information in server set SC64 is all at the same level. Thus, when evaluated in a packet this information will be ORed. This can be viewed as:

```
IF (Port 20) OR (Port 21) OR (Port 23) THEN Action
```

Now let us look at a Condition Set with multiple condition levels, which requires the AND function (see Figure 6-27 on page 150).



Condition Set	Server Set	Client Set	Application Set	Protocol Set	Outbound Address Set
Sample 1	SC64 servers			TCP	

Figure 6-27 zQoS ANDed condition

Using CNF, Condition Set sample 1 reads as:

```
IF SC64 Servers & TCP Then Action
```

Where:

TCP = TCP Protocol
SC64 Servers = Port 20 OR Port 21 or Port 23

Thus, making the appropriate substitutions we have:

```
If (TCP Protocol) AND (Port 20 OR Port 21 OR Port 23)
```

If we were to now include Condition Set sample 1 into a policy called test policy then the login would be:

```
Test Policy1:
If (TCP Protocol) AND (Port 20 OR Port 21 or Port 23) Then Action
```

6.6 Limitations

The zIDS Manager TCP/IP connection with the LDAP Server cannot be a secure connection, whereas the connection between PAGENT and the LDAP Server can be secure. IBM has been notified of this limitation via the newsgroup.

7

Implementing DiffServ policies

The best way to understand how to design, configure, and deploy QoS policies to meet your business needs is to examine the details in the context of a scenario. In this chapter we present a sample network to illustrate these concepts. To do so, we analyze our business requirements, create a set of traffic classes, and assign the classes to policies that relate to the business criteria.

We look at how the capabilities of the z/OS Policy Agent can be combined with the QoS features of the Cisco IOS to enable end-to-end Differentiated Services.

We examine the results of generated traffic sent over the QoS-enabled network in our lab environment and illustrate our policies in action.

While RSVP could also be defined and used for QoS signaling, we configured only DiffServ in our environment, since that is most applicable to a majority of enterprises.

7.1 Implementation steps

This section walks you through our sample scenario and illustrates how to develop a plan for QoS deployment, analyze traffic requirements, and assign classes and policies. As we will see, your policies will be directly influenced by the configuration of resources in your network as well as the types of application data flowing through them.

7.1.1 Perform traffic audit

As previously described in Chapter 5, “Understanding Quality of Service” on page 101, the first step in deploying QoS in a network is to analyze the application and traffic requirements. To adequately perform this step, it is necessary to have a good understanding of the current network environment. Figure 7-1 illustrates the network used in our lab scenario.

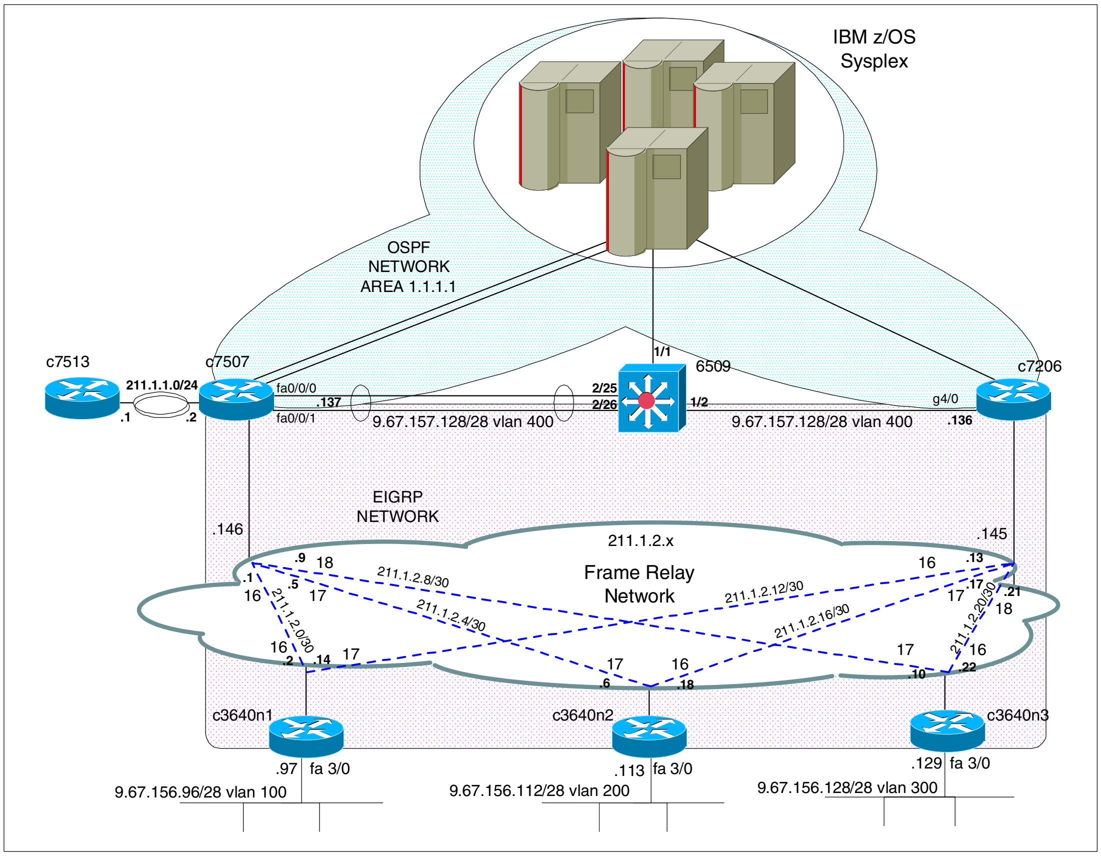


Figure 7-1 Our network environment

In our network, we have three remote sites connected via a frame relay network. Access to the sysplex is available through a Cisco 7507 and a Cisco 7206 VXR router. Each serial interface to the frame relay network operates at a speed of 1024 kbps. Permanent virtual circuits (PVCs) connect each of the remote sites to both sysplex ingress routers and have a defined committed information rate (CIR) of 256 kbps.

As part of the traffic audit, probes should be placed at various traffic aggregation points to analyze the traffic patterns present within the network. Armed with detailed information about the traffic types and patterns that exist within the network, the traffic audit can proceed to determine the associated applications and relative business priority for each traffic category. The results of our traffic audit are summarized in Table 7-1 on page 153.

Table 7-1 Traffic audit results

Application	Business priority	Traffic type
Order entry, customer service	1	SNA interactive
CRM	2	Telnet
HR	3	HTTP
File transfer	4	SNA batch
File transfer	4	FTP
Uncategorized	5	Other

Our sample network consists of several types of application traffic. To illustrate the support for QoS, we have identified Telnet, FTP, HTTP or Web traffic, SNA Interactive, and SNA Batch traffic types. We ranked the traffic in terms of business priority.

Another thing to consider in the traffic audit is any new application to be deployed in the network. For example, voice traffic is not part of our sample IP-based network today. It is anticipated that Voice over IP (VoIP) will be deployed in the future throughout our network. As a result, planners must take this into consideration when assigning classifications as outlined in 7.1.2, “Traffic classification” on page 153.

7.1.2 Traffic classification

Using the information from the traffic audit, we determine exactly what traffic classifications we will include in the network. We associate traffic classes with each of the identified traffic types according to the assigned business priority, as shown in Table 7-2. The goal is to minimize the number of classes and allocate traffic into aggregate classes.

Table 7-2 Traffic classification values

Application	Business priority	Traffic type	Class
Future voice	1	VoIP	LLQ
Order entry, customer service	1	SNA interactive	Platinum
CRM	2	Telnet	Gold
HR	3	HTTP	Silver
File transfer	4	SNA batch	Bronze
File transfer	4	FTP	Bronze
Uncategorized	5	Other	Best-effort

To meet our requirements, we define four general traffic classifications using the Olympic model: Platinum, gold, silver, and bronze. Below these service levels is the standard, best-effort service provided by IP. Low Latency Queuing (LLQ) can also be configured above platinum for voice traffic. Even though voice traffic may not currently be part of your network, we suggest refraining from allocating any other high-priority data traffic to this top priority.

7.1.3 QoS policy definition

We now assign and document the QoS policies that guide our actual configuration throughout the network, edge routers and switches, and host components. Figure 7-2 illustrates the ingress points where traffic marking takes place, and the congestion points where queuing and congestion management occurs.

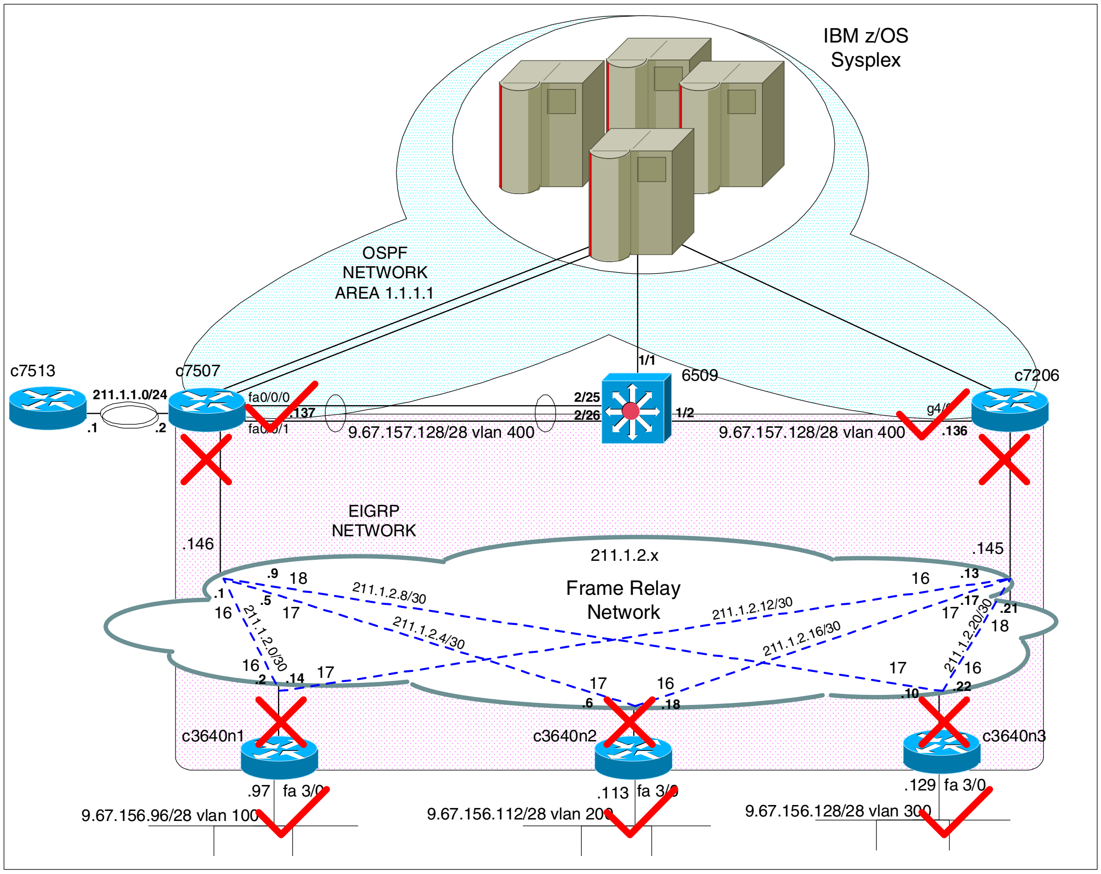


Figure 7-2 Congestion and re-marking points

Traffic is classified at the edges of the network. In our case, classification takes place at the 7507 and 7206 routers for traffic coming from the sysplex and at the 3640 routers for traffic originating at the remote LANs. These points are marked by check marks in the diagram.

In our simple network, the major congestion points are at the wide area network boundaries, marked by X in the diagram. It is at these points where Class-Based Weighted Fair Queuing (CBWFQ) will manage congestion and ensure applications are allocated bandwidth consistent with our business policies.

Voice traffic is not currently served by the IP network but is sure to be added in the future. By its nature, it requires consistent, low-latency service so the class-based priority queue; Low Latency Queuing (LLQ), will be reserved for voice. The other classes will be assigned relative bandwidth percentages.

Standard assured forwarding and expedited forwarding DSCP values are used for compatibility with carrier service networks and consistency across management domains. The DCSP values used are depicted in Table 7-3 on page 155.

Table 7-3 DSCP values

Class	Bandwidth	DSCP value	DSCP setting
VoIP	100k or 10 percent	46	101110, EF
Platinum	40 percent	10	001010, AF11
Gold	30 percent	12	001100, AF12
Silver	10 percent	18	010010, AF21
Bronze	5 percent	26	011010, AF31
Other	N/A	0	000000, or

7.2 Configuration examples

This section shows how we configured for QoS in our network.

7.2.1 z/OS configuration

The z/OS Policy Agent (PAGENT) was configured to mark packets according to the traffic classes we defined. Example 7-1 shows the configuration statements used in the PAGENT configuration file.

Example 7-1 PAGENT configuration

```
##################################################################
# Traffic Priorities for Policy Agent - Defaults to All interfaces
##################################################################

PolicyAction      DFSRV10
 {
   PolicyScope                            DataTraffic
   OutgoingTOS                            00101000  ;Bit Mask for TOS
 }
PolicyAction      DFSRV12
 {
   PolicyScope                            DataTraffic
   OutgoingTOS                            00110000  ;Bit Mask for TOS
 }
PolicyAction      DFSRV18
 {
   PolicyScope                            DataTraffic
   OutgoingTOS                            01001000  ;Bit Mask for TOS
 }
PolicyAction      DFSRV26
 {
   PolicyScope                            DataTraffic
   OutgoingTOS                            01101000  ;Bit Mask for TOS
 }

##################################################################
# Allows all IP interfaces to use the Policy Agent (0.0.0.0)
##################################################################

#-----------------------------------------------------------------
# Rules for SNA Batch and FTP
#-----------------------------------------------------------------
```

```
PolicyRule  DFSRV26_Rule_EE_Batch
 {
   Direction               Both
   PolicyScope             Both
   ProtocolNumber          UDP
   SourceAddressRange      9.67.156.1 9.67.156.1
   SourcePortRange         12004 12004
   ServiceReference        DFSRV26      # Service Catagory
 }

PolicyRule  DFSRV26_Rule_FTP
 {
   Direction               Both
   PolicyScope             Both
   SourcePortRange         20 21
   ServiceReference        DFSRV26      # Service Catagory
 }

#-----------------------------------------------------------------
# Rules for HTTP
#-----------------------------------------------------------------

PolicyRule  DFSRV18_Rule_Base_Web
 {
   Direction               Both
   PolicyScope             Both
   SourcePortRange         80     80    # HTTP Default Port
   PolicyActionReference   DFSRV18      # Service Catagory
 }

PolicyRule  DFSRV18_Rule_Secure_Web
 {
   Direction               Both
   PolicyScope             Both
   SourcePortRange         443    443   # HTTP Secure Port
   PolicyActionReference   DFSRV18      # Service Catagory
 }

#-----------------------------------------------------------------
# Rules for TELNET
#-----------------------------------------------------------------

PolicyRule  DFSRV12_Rule_Telnet
 {
   Direction               Both
   PolicyScope             Both
   SourcePortRange         23     23    # TELNET Default Port
   PolicyActionReference   DFSRV12      # Service Catagory
 }

PolicyRule  DFSRV12_Rule_Telnet_Secure
 {
   Direction               Both
   PolicyScope             Both
   SourcePortRange         523    523   # TELNET Default Port
   PolicyActionReference   DFSRV12      # Service Catagory
 }

#-----------------------------------------------------------------
# SNA Interactive
```

```
#----------------------------------------------------------------

PolicyRule  DFSRV10_Rule_EE_Interactive
 {
   Direction              Both
   PolicyScope            Both
   ProtocolNumber         UDP
   SourceAddressRange     9.67.156.1 9.67.156.1
   SourcePortRange        12002 12002
   ServiceReference       DFSRV10      # Service Catagory
 }
```

The outgoing TOS statement is used to set the various DSCP values for packets according to our policy criteria. The policyRule statements establish a means to distinguish the different traffic classes using a combination of IP addresses and port numbers. Each rule is tied back to the PolicyAction name with the ServiceReference statement.

SNA traffic being transported using Enterprise Extender (EE) is assigned to specific ports according to the SNA class of service and is always sourced from the static virtual IP address. This provides a convenient means to classify and mark this traffic. For example, the rule DFSRV10_Rule_EE_Interactive designates source IP address 9.67.156.1 and port 12002. This port is used for high-priority traffic using SNA COS #INTER.

7.2.2 Cisco network configuration

The remaining configuration examples show exactly how the Cisco network uses QoS to meet the end-to-end requirements developed in our sample scenario. First, the classes are defined in each of the routers. The commands in Example 7-2 define the classes.

Example 7-2 Cisco router class definitions

```
class-map match-all VoIP-EF
  description Voice traffic Expedited Forwarding
  match access-group 121
class-map match-all BestEffort-AF4
  description Default
  match access-group 114
class-map match-all Platinum-AF1
  description SNA Interactive
  match ip dscp 10
class-map match-any Silver-AF2
  description HTTP
  match ip dscp 18
  match access-group 112
class-map match-any Bronze-AF3
  description FTP, SNA Batch
  match ip dscp 26
  match access-group 113
class-map match-any Gold-AF1
  description Telnet
  match ip dscp 12
  match access-group 111
```

Access control lists are coupled with match conditions to classify the traffic. Match conditions that match on DiffServ DSCP values will classify traffic that is marked at trusted points in the network. In our case, the DSCP is set for traffic coming from the sysplex by PAGENT on the host. For example, PAGENT will have marked SNA interactive packets associated with the customer service and order entry applications with a DSCP value of b’00101000’. The first 6 bits of the DS field contain this DSCP value, b’001010’ = decimal 10. This matches the class-map Platinum-AF1. Notice that whenever you want an OR condition, use the match-any keyword rather than the match-all keyword. The access control lists associated with the access groups above are shown in Example 7-3.

Example 7-3 Access control lists

```
access-list 111 permit tcp any any eq telnet
access-list 112 permit tcp any any eq www
access-list 113 permit tcp any any eq ftp
access-list 114 permit ip any any
access-list 121 permit udp any any range 16384 32768
```

Next, a policy is applied for each of the traffic classes using the **policy-map** configuration command, as illustrated in Example 7-4.

Example 7-4 Policy mapping

```
policy-map FRpolicy
  class VoIP-EF
    priority 32
  class Platinum-AF1
   bandwidth percent 40
  class Gold-AF1
   bandwidth percent 30
  class Silver-AF2
   bandwidth percent 10
   random-detect
  class Bronze-AF3
   bandwidth percent 5
   random-detect
  class BestEffort-AF4
   bandwidth percent 5
```

The FRpolicy policy map, when applied to the frame relay interface, configures CBWFQ and determines the relative bandwidth settings for the traffic classes. However, it is important to make sure we consider the CIR associated with the PVCs that connect each site to the hub routers. By using traffic shaping we can be assured that our data traffic does not accumulate within the frame relay network and is instead buffered at the router and injected according to our QoS definition.

Example 7-5 Traffic shaping policy

```
policy-map FRshape
  class class-default
   shape average 256000
   service-policy FRpolicy
```

The statement `shape average 256000` indicates that Generic Traffic Shaping (GTS) is used and the average rate is limited to the committed information rate of 256000 bps. The FRshape policy map will be applied to the frame relay sub-interfaces and it will, in turn, specify the policy FRpolicy to apply the desired bandwidth percentages. Configured this way, it is, in effect, a nested or hierarchical policy definition.

The policy, FRshape, is shown applied to the frame relay sub-interfaces in Example 7-6.

Example 7-6 Generic traffic shaping

```
interface Serial5/0
 description NIVT Interoperability Frame Relay Cloud
 mtu 2106
 bandwidth 1024
 no ip address
 encapsulation frame-relay IETF
 no ip mroute-cache
 fair-queue 64 64 0
!
interface Serial5/0.1 point-to-point
 ip address 211.1.2.13 255.255.255.252
 service-policy output FRshape
 frame-relay interface-dlci 16
!
interface Serial5/0.2 point-to-point
 ip address 211.1.2.17 255.255.255.252
 service-policy output FRshape
 frame-relay interface-dlci 17
!
interface Serial5/0.3 point-to-point
 bandwidth 256
 ip address 211.1.2.21 255.255.255.252
 service-policy output FRshape
 frame-relay interface-dlci 18
```

We used the above configuration during our lab testing and achieved results we desired in our particular setup. However, we recommend that you use a configuration that specifies Frame Relay Traffic Shaping (FRTS). A preferred sample configuration using FRTS is shown in Example 7-7.

Example 7-7 FRTS sample configuration

```
interface Serial5/0
 description NIVT Interoperability Frame Relay Cloud
 mtu 2106
 bandwidth 1024
 no ip address
 encapsulation frame-relay IETF
 no ip mroute-cache
 frame-relay traffic-shaping
 !
interface Serial5/0.1 point-to-point
 ip address 211.1.2.13 255.255.255.252
 frame-relay interface-dlci 16
  class ts-class
!
interface Serial5/0.2 point-to-point
 ip address 211.1.2.17 255.255.255.252
 frame-relay interface-dlci 17
  class ts-class
!
interface Serial5/0.3 point-to-point
 bandwidth 256
 ip address 211.1.2.21 255.255.255.252
 frame-relay interface-dlci 18
  class ts-class
!
map-class frame-relay ts-class
 no frame-relay adaptive-shaping
 service-policy output FRshape
 frame-relay fragment 320
```

Example 7-7 on page 159 also shows how to enable fragmentation using the **frame-relay fragment** command. In this example we specify a fragment size of 320. This means that each fragment except the last will contain 320 bytes of the original payload. Fragmentation reduces the delay and jitter that might affect sensitive applications such as Voice over IP.

The policy map shown in Example 7-8, SETDSCP, when applied to the network ingress points, classifies and confirms the proper DSCP settings by re-marking.

Example 7-8 Policy to set DSCP

```
policy-map SETDSCP
  class VoIP-EF
   set ip dscp 46
  class Platinum-AF1
   set ip dscp 10
  class Gold-AF1
   set ip dscp 12
  class Silver-AF2
   set ip dscp 18
  class Bronze-AF3
   set ip dscp 26
```

The SETDSCP policy is applied to the ingress interface using the **service-policy** command in Example 7-9. Note the keyword *input* is specified to define it as an input policy.

Example 7-9 Applying the input policy for classification

```
interface GigabitEthernet4/0
 ip address 9.67.157.136 255.255.255.240
 ip igmp join-group 224.0.1.2
 negotiation auto
 service-policy input SETDSCP
```

As mentioned earlier, in cases where class-based packet marking cannot be applied to the interface (on the 7507 CIP channel interface, for example), we rely on the host-based Policy Agent to mark the packets. Close coordination of definitions between those applied by the systems programmer and the network administrator is required.

On the remote 3640 routers, SNA traffic is handled by Cisco SNA Switching Services (SNASw). SNASw, configured for Enterprise Extender, transmits SNA traffic as UDP packets sourced from a loopback interface. Class-Based Packet Marking for router-generated traffic is not supported until Cisco IOS 12.2(4). Since we are running Cisco IOS 12.2(1a), we used the **police** command to ensure the marking of DSCP according to our chosen scheme.

Additional ACLs identify traffic using the source loopback interface IP address and the COS-related IP precedence bits set by SNASw, as shown in Example 7-10.

Example 7-10 Additional access control lists

```
access-list 115 permit udp host 9.67.157.5 any precedence flash-override
access-list 115 permit udp host 9.67.157.5 any precedence immediate
access-list 115 permit udp host 9.67.157.9 any precedence network
access-list 116 permit udp host 9.67.157.5 any precedence priority
```

These lists are added to the classification using the keyword match-any on the class-map, as shown in Example 7-11.

Example 7-11 Adding the ACLs to classifications

```
class-map match-any Platinum-AF1
  description SNA Interactive
  match ip dscp 10
  match access-group 115

class-map match-any Bronze-AF3
  description FTP, SNA Batch
  match ip dscp 26
  match access-group 113
  match access-group 116
```

Note the use of the match-any condition rather than match-all. This makes the condition that of a logical OR rather than an AND for the class.

The **police** command is then used to mark the packets accordingly, as indicated in Example 7-12. In our case, we were not concerned with policing the traffic, but only ensuring the packets were marked with the correct DSCP value.

Example 7-12 Using the police command

```
policy-map FRpolicy
  class VoIP-EF
    priority 32
  class Platinum-AF1
   bandwidth percent 40
     police 512000 5000 5000 conform-action set-dscp-transmit 10 exceed-action
set-dscp-transmit 10 violate-action set-dscp-transmit 10
  class Gold-AF1
   bandwidth percent 30
  class Silver-AF2
   bandwidth percent 10
   random-detect
  class Bronze-AF3
   bandwidth percent 5
   random-detect
     police 62000 2000 2000 conform-action set-dscp-transmit 26 exceed-action
set-dscp-transmit 26 violate-action set-dscp-transmit 26
  class BestEffort-AF4
   bandwidth percent 5
```

Verification and confirmation of the effects of the policy can be seen using the **show policy interface** command. The match on access-list 115 causes the classification of SNASw-originated traffic as Platinum-AF1 and the policy assures the UDP packets transmitted over the frame relay interface are marked with DSCP value 10. Note the packet count under the Platinum-AF1 class and the conformed packet count and action under the police heading, shown in Example 7-13.

Example 7-13 Show policy interface command

```
C3640N2#sh pol int

* Some output deleted *

Serial1/1.2

  Service-policy output: FRshape

    Class-map: class-default (match-any)
      74465 packets, 6373664 bytes
      5 minute offered rate 11000 bps, drop rate 0 bps
      Match: any
      Traffic Shaping
```

```
        Target    Byte   Sustain   Excess    Interval  Increment Adapt
        Rate      Limit  bits/int  bits/int  (ms)      (bytes)   Active
        256000    1984   7936      7936      31        992       -

        Queue     Packets   Bytes     Packets   Bytes     Shaping
        Depth                         Delayed   Delayed   Active
        0         69937     6010446   978       303375    no

* Some output deleted *

        Class-map: Platinum-AF1 (match-any)
          1014 packets, 179623 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: ip dscp 10
            0 packets, 0 bytes
            5 minute rate 0 bps
          Match: access-group 115
            1014 packets, 179623 bytes
            5 minute rate 0 bps
          Weighted Fair Queuing
            Output Queue: Conversation 41
            Bandwidth 40 (%) Max Threshold 64 (packets)
            (pkts matched/bytes matched) 10/2145
        (depth/total drops/no-buffer drops) 0/0/0
          police:
            512000 bps, 5000 limit, 5000 extended limit
            conformed 1015 packets, 179722 bytes; action: set-dscp-transmit 10
            exceeded 0 packets, 0 bytes; action: set-dscp-transmit 10
            violated 0 packets, 0 bytes; action: set-dscp-transmit 10
            conformed 0 bps, exceed 0 bps violate 0 bps
```

Class-based packet marking is being enhanced with Cisco IOS 12.2(4) to allow for marking of router-generated packets such as those originated from SNASw. This will facilitate class-based marking without the use of class-based policing as was used here.

7.3 QoS test results

In order to show the effects of QoS and the interaction between the host-based PAGENT and the QoS features configured in the network, a traffic generator tool was used to simulate the application data.

We used a tool called Chariot to send data among various endpoints at the remote sites and the host. Specific port numbers were used to simulate traffic categories, which were marked using host-based PAGENT. The traffic profiles for each class were normalized so that the effects of QoS would be apparent. In other words, the same traffic profile was used for each traffic class.

The effects of PAGENT and verification of the defined policies can be seen using the **netstat slap** command on the host, as shown in Example 7-14.

Example 7-14 PAGENT verification

```
===> netstat slap

MVS TCP/IP NETSTAT CS V1R2       TCPIP NAME: TCP              20:36:16,
PolicyRuleName:  DFSRV10_Rule_EE_Interactive,
  FirstActTime:    20:15:55      LastMapTime:      20:17:24,
  TotalBytesIn:    0000000000    TotalBytesOut:    0001409598,
  BytesInDiscard:  0000000000    BytesOutDiscard:  0000000000,
  TotalInPackets:  0000000000    TotalOutPackets:  0000002054,
  ActConnMap:      0000000000    MaxConnLimit:     0000000000,
  AcceptConn:      0000000000    DeniedConn:       0000000000,
  OutBytesInProf:  0000000000    Status:           Active,
PolicyRuleName:  DFSRV26_Rule_EE_Batch,
  FirstActTime:    20:15:55      LastMapTime:      20:17:24,
  TotalBytesIn:    0000000000    TotalBytesOut:    0000883228,
  BytesInDiscard:  0000000000    BytesOutDiscard:  0000000000,
  TotalInPackets:  0000000000    TotalOutPackets:  0000000999,
  ActConnMap:      0000000006    MaxConnLimit:     0000000000,
  AcceptConn:      0000000075    DeniedConn:       0000000000,
  OutBytesInProf:  0000000000    Status:           Active,
PolicyRuleName:  DFSRV12_Rule_Telnet_Secure,
  FirstActTime:    20:15:55      LastMapTime:      00:00:00,
  TotalBytesIn:    0000000000    TotalBytesOut:    0000000000,
  BytesInDiscard:  0000000000    BytesOutDiscard:  0000000000,
  TotalInPackets:  0000000000    TotalOutPackets:  0000000000,
  ActConnMap:      0000000006    MaxConnLimit:     0000000000,
  AcceptConn:      0000000615    DeniedConn:       0000000000,
  OutBytesInProf:  0000000000    Status:           Active,
PolicyRuleName:  DFSRV12_Rule_Telnet,
  FirstActTime:    20:15:55      LastMapTime:      20:35:40,
  TotalBytesIn:    0000243516    TotalBytesOut:    0005321980,
  BytesInDiscard:  0000001660    BytesOutDiscard:  0000000000,
  TotalInPackets:  0000004898    TotalOutPackets:  0000005725,
  ActConnMap:      0000000006    MaxConnLimit:     0000000000,
  AcceptConn:      0000000615    DeniedConn:       0000000000,
  OutBytesInProf:  0000000000    Status:           Active,
PolicyRuleName:  DFSRV18_Rule_Secure_Web,
  FirstActTime:    20:15:55      LastMapTime:      00:00:00,
  TotalBytesIn:    0000000000    TotalBytesOut:    0000000000,
  BytesInDiscard:  0000000000    BytesOutDiscard:  0000000000,
  TotalInPackets:  0000000000    TotalOutPackets:  0000000000,
  ActConnMap:      0000000006    MaxConnLimit:     0000000000,
  AcceptConn:      0000000619    DeniedConn:       0000000000,
  OutBytesInProf:  0000000000    Status:           Active,
PolicyRuleName:  DFSRV18_Rule_Base_Web,
  FirstActTime:    20:15:55      LastMapTime:      20:35:40,
  TotalBytesIn:    0000278896    TotalBytesOut:    0002639496,
  BytesInDiscard:  0000001220    BytesOutDiscard:  0000000000,
  TotalInPackets:  0000003155    TotalOutPackets:  0000003311,
  ActConnMap:      0000000006    MaxConnLimit:     0000000000,
  AcceptConn:      0000000619    DeniedConn:       0000000000,
  OutBytesInProf:  0000000000    Status:           Active,
PolicyRuleName:  DFSRV26_Rule_FTP,
  FirstActTime:    20:15:55      LastMapTime:      20:36:10,
  TotalBytesIn:    0000062636    TotalBytesOut:    0003879069,
  BytesInDiscard:  0000001796    BytesOutDiscard:  0000000000,
  TotalInPackets:  0000001900    TotalOutPackets:  0000002821,
  ActConnMap:      0000000006    MaxConnLimit:     0000000000,
  AcceptConn:      0000000075    DeniedConn:       0000000000,
  OutBytesInProf:  0000000000    Status:           Active,
```

Note that in Example 7-14 on page 162 the packet counts in each of the traffic classes we defined: EE_Interactive, EE_Batch, Telnet, Base_Web, and FTP. The PAGENT registered our traffic sent over our test network in the categories as we expected.

The following examples show the effects of QoS when applied to the data center serial interfaces.

We first show response time for various traffic categories without QoS applied. To do so, QoS policies were removed from the configuration at the congestion point where outbound traffic enters the frame relay network, as depicted in Example 7-15. The serial link on the 7507 was shut down so that all traffic traveled over the link attached to the 7206 and the QoS effects would be seen clearly.

Example 7-15 QoS policy removed

```
int s5/0.1
no service-policy output FRshape
int s5/0.2
no service-policy output FRshape
int s5/0.3
no service-policy output FRshape
```

The output from the **`show interface`** command on the router confirms the lack of queuing set on the interface. Note the `Queuing strategy: fifo` line in Example 7-16. This indicates that only best-effort, first-in-first-out queuing is in effect.

Example 7-16 Show interface command after QoS is disabled

```
C7200-Z55#sh in s5/0
Serial5/0 is up, line protocol is up
  Hardware is M4T
  Description: NIVT Interoperability Frame Relay Cloud
  Internet address is 9.67.156.145/28
  MTU 2106 bytes, BW 256 Kbit, DLY 20000 usec,
     reliability 255/255, txload 242/255, rxload 23/255
  Encapsulation FRAME-RELAY IETF, crc 16, loopback not set
  Keepalive set (5 sec)
  LMI enq sent  300, LMI stat recvd 300, LMI upd recvd 0, DTE LMI up
  LMI enq recvd 0, LMI stat sent  0, LMI upd sent  0
  LMI DLCI 0  LMI type is ANSI Annex D  frame relay DTE
  FR SVC disabled, LAPF state down
  Broadcast queue 0/64, broadcasts sent/dropped 153/0, interface broadcasts 26
  Last input 00:00:02, output 00:00:00, output hang never
  Last clearing of "show interface" counters 00:25:01
  Queuing strategy: fifo
  Output queue 0/150, 459 drops; input queue 0/75, 0 drops
  30 second input rate 25000 bits/sec, 32 packets/sec
  30 second output rate 258000 bits/sec, 34 packets/sec
     14655 packets input, 1364931 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     14917 packets output, 11302859 bytes, 0 underruns
     0 output errors, 0 collisions, 1 interface resets
     0 output buffer failures, 0 output buffers swapped out
     1 carrier transitions DCD=up  DSR=up  DTR=up  RTS=up  CTS=up
```

The graph in Figure 7-3 on page 165 was generated by Chariot and shows response time for each traffic category.

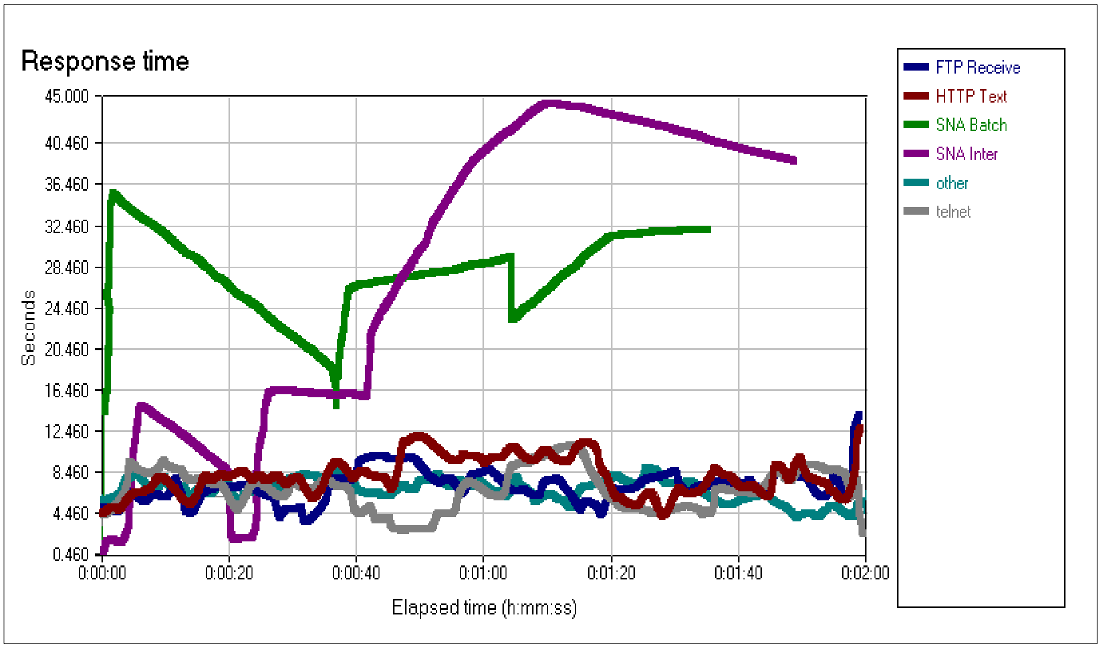


Figure 7-3 Response time without QoS in the network

It can be seen from the chart that SNA traffic (top two lines) is almost starved out by the competing IP data. The FIFO queuing offers no benefit to any particular traffic category. This provides an example of what can happen in a network without any regard to traffic classification or prioritization. Compare these results with those following application of the QoS policies.

The QoS policies were re-applied to the outbound congestion point, as shown in Example 7-17.

Example 7-17 QoS policy applied

```
int s5/0.1
service-policy output FRshape
int s5/0.2
service-policy output FRshape
int s5/0.3
service-policy output FRshape
```

The output from the **show interface** command on the router confirms application of the policy and the active queuing strategy, as shown in Example 7-18.

Example 7-18 Show interface command after QoS is enabled

```
C7200-Z55#sh in s5/0
Serial5/0 is up, line protocol is up
  Hardware is M4T
  Description: NIVT Interoperability Frame Relay Cloud
  Internet address is 9.67.156.145/28
  MTU 2106 bytes, BW 256 Kbit, DLY 20000 usec,
     reliability 255/255, txload 251/255, rxload 14/255
  Encapsulation FRAME-RELAY IETF, crc 16, loopback not set
  Keepalive set (5 sec)
  LMI enq sent  626, LMI stat recvd 626, LMI upd recvd 0, DTE LMI up
  LMI enq recvd 0, LMI stat sent  0, LMI upd sent  0
  LMI DLCI 0  LMI type is ANSI Annex D  frame relay DTE
  FR SVC disabled, LAPF state down
  Broadcast queue 0/64, broadcasts sent/dropped 321/0, interface broadcasts 55
  Last input 00:00:03, output 00:00:00, output hang never
  Last clearing of "show interface" counters 00:52:12
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 825
  Queuing strategy: weighted fair
  Output queue: 0/150/64/0 (size/max total/threshold/drops)
     Conversations  0/1/64 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 192 kilobits/sec
  30 second input rate 15000 bits/sec, 27 packets/sec
  30 second output rate 252000 bits/sec, 26 packets/sec
     23501 packets input, 2028921 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     23659 packets output, 16517826 bytes, 0 underruns
     0 output errors, 0 collisions, 2 interface resets
     0 output buffer failures, 0 output buffers swapped out
     2 carrier transitions     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up

C7200-Z55#
```

The graph in Figure 7-4 shows the response time for the various traffic categories with QoS applied in the network.

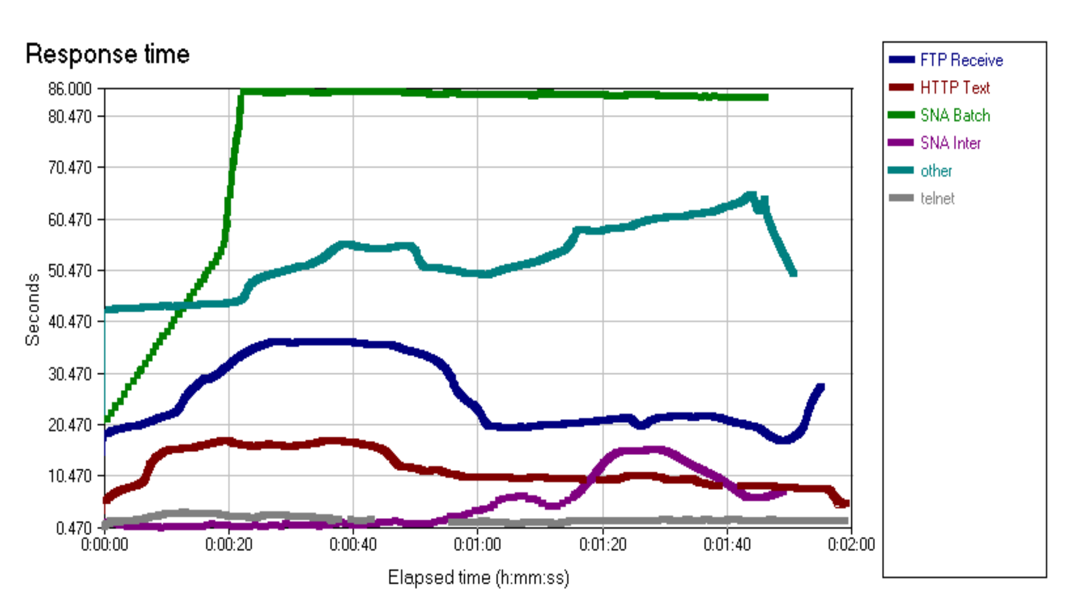


Figure 7-4   Response time with QoS enabled in the network

The effects of the QoS policies can be clearly seen in the results. Traffic classes incur response times corresponding to their relative bandwidth allocation. Response time for SNA interactive traffic, for example, averaged 1.57 seconds with QoS enabled, as opposed to 10.95 seconds without QoS enabled, as illustrated in Table 7-4.

Table 7-4   Results of QoS policies on response time

Class	**Average response time with QoS**	**Average response time without QoS**
FTP receive	22.83638	4.9296

Class	Average response time with QoS	Average response time without QoS
HTTP text	10.14838	4.97382
SNA batch	53.279	20.7397
SNA interactive	1.57708	10.9572
Telnet	1.75396	4.35252
Other	47.17735	4.58654

7.3.1 Summary

This scenario illustrates how to create and use the QoS Policy Agent on the host in combination with the QoS features available in Cisco IOS to achieve service policies relative to business criteria.

While the information was presented using a scenario in our lab environment rather than an actual, enterprise customer deployment, the concepts and procedures apply to many enterprises and service providers that attempt to deploy end-to-end Quality of Service.

8

Implementing Sysplex Distributor policies

Sysplex Distributor was introduced in IBM Communications Server for OS/390 V2R10 IP and it took the XCF dynamics and dynamic VIPA support to a whole new level in terms of availability and workload balancing in a sysplex. Selected workload can be distributed to multiple server instances within the sysplex without requiring changes to clients or networking hardware and without delays in connection setup. IBM Communications Server for z/OS V1R2 provides the way to implement a dynamic VIPA as a single network-visible IP address for a set of hosts that belong to the same sysplex cluster. Any client located anywhere in the IP network is able to see the sysplex cluster as one IP address regardless of the number of hosts that it includes.

With Sysplex Distributor, clients receive the benefits of workload distribution provided by both Workload Manager (WLM) and Quality of Service (QoS) policies through the Policy Agent. In addition, Sysplex Distributor ensures high availability of the IP applications running on the sysplex cluster, no matter if one physical network interface fails or an entire IP stack or z/OS host is lost.

This chapter includes:

- A high-level overview of Sysplex Distributor
- Sysplex Distributor and policy

8.1 What a Sysplex Distributor is

Sysplex Distributor was designed to address the requirement of one single network-visible IP address for the sysplex cluster and let the clients in the network receive the benefits of workload distribution and high availability within the sysplex cluster. With Sysplex Distributor, client connections seem to be connected to a single IP host even if the connections are established with different servers in the same sysplex cluster.

Because the Sysplex Distributor function resides on a system in the sysplex itself, it has the ability to factor "real-time" information concerning the multiple server instances including server status as well as QoS and policy information provided by CS for z/OS IP's Policy Agent. By combining these real-time factors with the information from WLM, the Sysplex Distributor has the unique ability to ensure that the best destination server instance is chosen for a particular client connection. The Sysplex Distributor has more benefits than other load-balancing implementations, such as the Network Dispatcher or DNS/WLM. Their limitations are removed with Sysplex Distributor.

In summary, the benefits of Sysplex Distributor include:

- Removes configuration limitations of Network Dispatcher.
- Target servers can use XCF links (or a hipersocket connection if lpars are on the same CEC) between the distributing stack and target servers as opposed to LAN connections such as an OSA.
- Removes dependency of specific hardware in WAN.
- Provides total CS for z/OS IP solution for workload distribution.
- Provides real-time workload balancing for TCP/IP applications, even if clients cache the IP address of the server (a common problem for DNS/WLM).
- Enhanced VIPA takeover and takeback support.
- Allows for non-disruptive takeback of VIPA original owner to get workload where it belongs.
- Distributing function can be backed up and taken over.
- Enhances Dynamic VIPA support.
- Non-disruptive application server instance movement.

In summary, Sysplex Distributor provides:

- Single network-visible IP address of a sysplex cluster service. One IP address can be assigned to the entire sysplex cluster (usually for each service provided, such as Telnet):
- Sysplex Distributor will query the Policy Agent to find if there exists any policy defined for routing the incoming connection requests.
- WLM and QoS policy can be specified for workload balancing in real-time on every new connection request.
- Up to 256 DVIPAs on a stack.
- Backup capability is enhanced. In case of failure of the distributing IP stack, the connections distributed to other IP stacks in the sysplex will not be disrupted.
- Dynamic VIPA takeback without any disruption in the connections already established.
- Commands to display both the connection routing table and destination port table information, showing information of configuration and current connection distribution.

For more detailed information on Sysplex Distributor please refer to *Communications Server for z/OS V1R2 TCP/IP Implementation Guide Volume 1: Base and TN3270 Configuration,* SG24-5227.

8.2 Sysplex Distributor and policy

Policies are an administrative means to define controls for a network in order to achieve the QoS levels promised by a given SLA or to implement security or resource balancing decisions. Quality of Service Policy allows classification of IP traffic by application, user group, time of day, and assignment of relative priority. The Policy Agent reads policy entries from a flat file called pagent.conf that can be located in any MVS or HFS file or from an LDAP Server or both. The Sysplex Distributor uses these policies to limit the target stack to route its work in conjunction with the WLM weights. Note that the WLM has to run in GOAL mode (which is the only WLM option when your system is at z/OS V1R3 or higher) at the target stacks, or the QoS weights will have no effect and the distribution of the work is random.

The following types of policies are supported in IBM Communication Server for z/OS:

Integrated Services Type of service that provides end-to-end QoS using resource reservations along a network path from sender to receiver. This service is provided by the RSVP Agent.

Differentiated Services Type of services that provides aggregate QoS to broad classes of traffic (for example, all FTP traffic).

Sysplex Distribution Policies specify to which target stack the Sysplex Distributor may route incoming connection requests.

Traffic Regulation Mgmt Policies define the maximum number of connections to a TCP port and control the number from a single host to this port.

The Policy Agent performs two distinct functions to assist the Sysplex Distributor:

1. Policies can be defined to control which stack the Sysplex Distributor routes traffic to. The definition of the outbound interface on the PolicyAction statement can limit the stacks to which work is distributed to a subset of those defined on the VIPADISTRIBUTE statement in the TCPIP.PROFILE. Using a policy, the stack to which work is distributed can vary, for example, based on time periods. Another possibility is to limit the number of SD target stacks for inbound traffic from a given subnet (Figure 8-1 on page 172).
2. The PolicyPerfMonitorForSDR statement in the pagent.conf file will activate the Policy Agent QoS performance monitor function. When activated, the Policy Agent will use data about packet loss and timeouts that exceeds defined thresholds and derive a QoS weight fraction for that target stack. This weight fraction is then used to reduce the WLM weight assigned to the target stacks so that the Sysplex Distributor stack can use this information to better direct workload to where network traffic is best being handled. This policy is activated on SD target stacks (Figure 8-1 on page 172).

Attention: Sysplex Distributor normalizes the weights it receives from WLM (divides all the WLM weights by the lowest valued of WLM weight and then rounds the result to the nearest interger. Therefore in a Sysplex, not under load, it is *not* unusual to see the WLM weights of all LPARS have a value of 1.

To exclude stale data from target stacks where the Policy Agent has terminated, the Policy Agent sends a “heartbeat” to the SD distributing stack at certain intervals. The SD distributing stack deletes QoS weight fraction data from a target stack when the heartbeat has not been received within a certain amount of time.

At ITSO Raleigh we configured SD policies in two ways. In one scenario, the Policy Agent extracts the policy information from a static configured HFS file (PAGENT file), and in another case, an LDAP Server running in OS/390 provides all policy information in the network.

The sysplex consists of three z/OS systems, SC63, SC64, and SC65. On each system, there is a TCP/IP stack named TCPIPB, which participates in Sysplex Distributor. The TCPIPB stack on SC64 takes the role of the distributing stack, and the stacks on SC63 and SC65 act as target stacks. A Telnet server is listening on port 2364 on all systems, and has been configured on each SD stack as an application served by SD.

We have defined an SD policy for Telnet connections to install into the SD distributing stack, which is TCPIPB on SC64, and have defined an SD performance monitoring policy for the SD target stacks, which are TCPIPB on SC63 and SC63. Figure 8-1 illustrates the system environment at ITSO Raleigh.

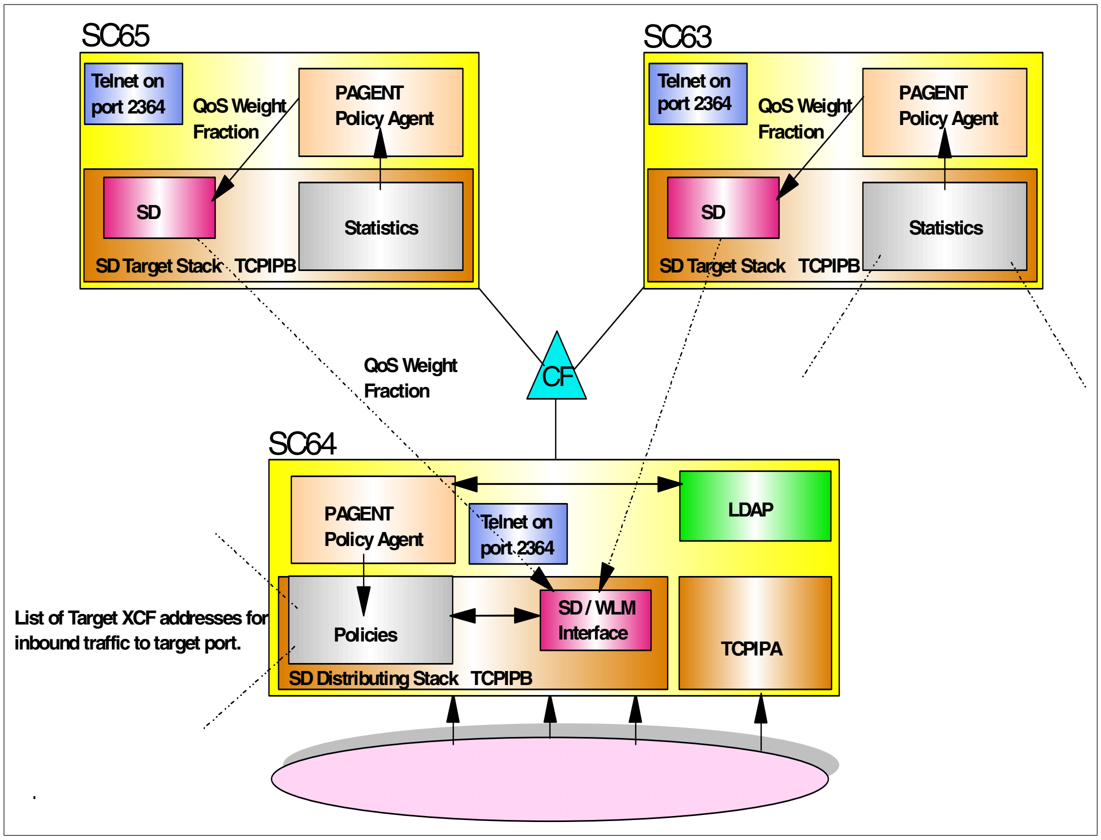


Figure 8-1 Sysplex Distributor policy implementation at ITSO Raleigh

8.2.1 Sysplex Distributor QoS policy in the PAGENT file

In our sysplex, the VIPADYNAMIC statements coded in the distributing stack, SC64, are as follows:

```
VIPADYNAMIC
   VIPADEFINE MOVEABLE IMMEDIATE 255.255.255.0 9.12.6.64
   VIPADISTRIBUTE DEFINE 9.12.6.64 PORT 2364
      DESTIP 10.1.1.6  10.1.1.4
ENDVIPADYNAMIC
```

We defined the SD policies that limit the number of SD target stacks for inbound traffic on the SD target stack, and the SD performance monitoring policies on all the participating stacks. For SD performance monitoring, the traffic to be monitored has to be represented by at least one Differentiated Services policy defined for the target application.

All policies were configured in the PAGENT's configuration file, etc/pagent.pok.conf, on the respective system.

You will find further information about SD policy and SD performance monitor policy in *z/OS V1R2 Communications Server: IP Configuration Reference*, SC31-8776, and *z/OS V1R2 Communications Server: IP Configuration Guide,* SC31-8775.

We have the following SD policy configured in the SD distributing stack (TCPIPB on SC64):

```
#
#  IBM Communications Server for z/OS
#  SMP/E distribution path: /usr/lpp/tcpip/samples/IBM/EZAPAGCO
#
#       /etc/pagent.pok.conf  (PAGENT policy configuration)
#
TcpImage TCPIPB FLUSH NOPURGE 1800
Loglevel 511
# PolicyPerfMonitorForSDR Statement
PolicyPerfMonitorForSDR enable
{
  samplinginterval 60
  LossRatioAndWeightFr 20 25
  TimeoutRatioAndWeightFr 50 50
  LossMaxWeightFr 95
  TimeoutMaxWeightFr 100
}
# Test distribution rules
PolicyRule telnetd # telnet traffic
{
# ProtocolNumberRange 6
  Direction Both
  DestinationPortRange 2364                   1
  PolicyActionReference interactivez          2
  ForLoadDistribution True                    3
}
PolicyAction interactivez                     2
{
  PolicyScope DataTraffic
  OutboundInterface 10.1.1.4                  4
  OutboundInterface 10.1.1.6                  4
# OutboundInterface 0.0.0.0                   5
}
```

The policies are identified as SD policies by the presence of the ForLoadDistribution 3 on the policyRule statement. The Outboundinterface 4 attribute in the PolicyAction statement identifies the IP addresses of the SD target stacks dynamic XCF interfaces that a packet matching this rule may be distributed to. Up to 32 instances of this attribute can be specified. Refer to Chapter 5 in *Communications Server for z/OS V1R2 Implementation Guide Volume 3: Availability, Scalability, and Performance,* SG24-6517, for more information on dynamic XCF links participating in Sysplex Distributor.

2 A value of zero can be specified for the interface, which indicates to the SD distributing stack that if it cannot distribute the request to a target stack on one of the other specified interfaces, then the request can be distributed to any of the other eligible target stacks.

Because we commented out the Outboundinterface 0.0.0.0 definition, SD will reject the request when neither of the target stacks is available.

In our implementation, the incoming Telnet connection requests to port 2364 will be forwarded to either SC63 or SC65, even though there is a Telnet server listening on port 2364 on SC64. Without this SD policy activated, an incoming Telnet connection to port 2364 will be forwarded to one of three systems.

Note: For FTP and other applications that use a control port and a data port, you always have to define both.

2 You always match the policyRule to the policyAction by the policyActionReference statement.

An additional possibility would be to activate this policy only at certain times, let us say during normal working hours from Monday to Friday between 08:00 and 17:00, which we did not do in our implementation. But you would have to add only two statements to the policyRule definitions. Check the sample file /usr/lpp/tcpip/samples/pagent.conf for the correct syntax and explanation.

```
DayOfWeekMask      0111110
TimeOfDayRange     08:00-17:00
```

On the SD target stacks (SC63 and SC65), the following policies have been activated:

```
#
#  IBM Communications Server for z/OS
#  SMP/E distribution path: /usr/lpp/tcpip/samples/IBM/EZAPAGCO
#
# LogLevel Statement
Loglevel 511

TcpImage TCPIPB FLUSH NOPURGE 1800

# PolicyPerfMonitorForSDR Statement
PolicyPerfMonitorForSDR  enable                       1
{
  samplinginterval 60                                 2
  LossRatioAndWeightFr 20 25                          3
  TimeoutRatioAndWeightFr 50 50                       4
  LossMaxWeightFr 95                                  5
  TimeoutMaxWeightFr 100                              6
}
# Test distribution rules
PolicyRule telnetd # telnet traffic                   9
{
# ProtocolNumberRange 6
  Direction Both
  SourcePortRange 2364                                10
  PolicyActionReference interactivez
}
PolicyAction interactivez                             7
{
  PolicyScope DataTraffic                             8
  MaxConnections 8       # Limit the number of Telnet connections on port 2364 to 8
}
```

1 Enables the policy performance monitor function, which assigns a weight fraction to the monitored policy performance data and sends it to the SD distributing stack as the monitored data crosses defined thresholds. The SD distributing stack uses this weight fraction for its routing decisions for incoming connection requests.

2 With the samplingInterval we specify how often we want to sample the policy information for changes.

3 The LossRatioAndWeightFr has two values: The ratio of retransmitted bytes over transmitted bytes in tenths of a percent, and the weight fraction to be assigned in percentage. In our implementation, if a loss ratio rate of 2 percent occurs, the loss fraction will be 25 percent. If a loss ratio rate above 4 percent occurs, the loss weight fraction will be 50 percent. Let us assume we have a loss ratio rate of 3 percent, which means a loss weight fraction of 25 percent.

4 The TimeoutRatioAndWeightFr has two values: The timeout ratio in tenths of a percent, and the weight fraction to be assigned in percentage. In our implementation, if a timeout ratio rate of 5 percent occurs, the timeout fraction weight would be 50 percent. If the timeout ratio rate is above 10 percent, the timeout weight fraction would be 100 percent. Let us assume we have a timeout ratio of 6 percent, which means we have a timeout weight fraction of 50 percent.

The two weight fractions LossRatioAndWeightFr and TimeoutRatioWeightFr will then be added. In our case, we would have a QoS weight fraction value of 75 percent, which this target stack sends to the Sysplex Distributor. This value will be used by the Sysplex Distributor stack to reduce the WLM weight given to this stack. So if the WLM weight assigned to this target stack is 50, the QoS weight fraction of 75 percent will give an effective WLM weight fraction of 37.5.

5 LossMaxWeightFr defines the maximum loss weight fraction.

6 TimeoutMaxWeightFr defines the maximum timeout weight fraction.

The Policy Agent monitors the traffic for which one or more Service Policy statements have been defined 7, 9. The policyScope attribute for the monitored policy has to be either DataTraffic 8 or Both.

Important: You will see that we have coded, in the policy rule on the target system, SourcePortRange 10 rather than DestinationPortrange, as we did in the distributing policy. The reason for this is that, on the target system, policies are only checked when the target stack is ready to respond. (Target system is looking out into the network.) On the distributing system policies are checked when they first arrive at the stack.

Policies on the distributing system when its is also a target

It is perfectly reasonable in your sysplex, for the distributing system to also be is also a potential target system for a connection; the XCF interface of the distributing system was coded on the VIPADISTRIBUTE DESTIP address. In our sysplex this would be 10.1.1.5 as follows:

```
VIPADISTRIBUTE DEFINE 9.12.6.64 PORT 2364
DESTIP 10.1.1.6  10.1.1.4 10.1.1.5
```

In the Policy Agent on the distributing stack you will need to add an new policy rule 1, specifying ForLoadDistribution False 2 (which indicates that this rule/action is not to be distributed). Add the XCF interface of the distributing stack to the OutboundInterface list 4. As the policy rule will only be checked when the now target stack is ready to respond, the rule should be written as such 3 (SourcePortRange rather than DestportRange). If you are using QoS service level fractions (see “Sysplex Distributor QoS Service Level Fractions” on page 176), you will be aware that the policy action names need to be the same across the sysplex. Therefore, you will need to include additional parameters that the policy action will invoke when the Action is not being distributed. In our case this was to add the Maxconnections parameter 5.

```
# Distributed Rule
PolicyRule telnetd # telnet traffic
{
# ProtocolNumberRange 6
  Direction Both
  DestinationPortRange 2364
  PolicyActionReference interactivez
  ForLoadDistribution True
# Rule for when the Distributor is also a Target
PolicyRule telnett  1  # telnet traffic
{
# ProtocolNumberRange 6
  Direction Both
  SourcePortRange 2364                       3
  PolicyActionReference interactivez
  ForLoadDistribution False                  2

}
PolicyAction interactivez
{
  PolicyScope DataTraffic
  OutboundInterface 10.1.1.4
  OutboundInterface 10.1.1.6
  OutboundInterface 10.1.1.5                 4
  MaxConnections 8                           5
}
```

Note: When a connection for a distributed DVIPA is received, TCP/IP first checks to see if there is a distributed Policy Rule. If there is, then it distributes the policy according to the policy action. If the connection is distributed to this system then TCP/IP will check and see if there is a non-distributed policy rule that matches this connection. If there is, then it will refer to the non-distribution part of the policy action.

8.2.2 Sysplex Distributor QoS Service Level Fractions

In Figure 8-1 on page 172, we showed Policy Agent as it is would be implemented in an OS/390 Sysplex. Policy Agents on the Target stacks collect QoS performance data for each DVIPA and port. This information would then be passed to the stacks Sysplex Distributor function, which in turn forwards it to the Sysplex Distributor function on Distributing stack.

At the z/OS V1R2 level you may continue to run in this fashion. However, CS for z/OS V1R2 has provided an additional level of granularity, such that Policy Agent running on target stacks will gather performance data for all connections that map to a policy action (service level) for each distributed VIPA and its associated port (DVIPA/port). Policy Agents on the Distributing system (pagentQosCollector) collect this information by sending on a Policy Agent - Policy Agent TCP connection (Figure 8-2 on page 177), a list of QoS service level names, to Policy Agents on the target systems (pagentQosListener). Policy Agent on each target will send back a QoS policy action weight fraction for each requested service level for each target DVIPA/PORT pair that it supports. Upon receiving the QoS policy action weight fraction, the Policy Agent on the distributing stack will pass this information to the Sysplex Distribution function on the stack. The stack Sysplex Distribution function will use this additional information when it selects targets for incoming connections. If it does not have a QoS policy action weight fraction, then it will use the existing weight fraction described above to make the load distribution decision.

> **Note:** A specific policy action weight fraction will not be sent unless the distributing stack's Policy Agent requests it.

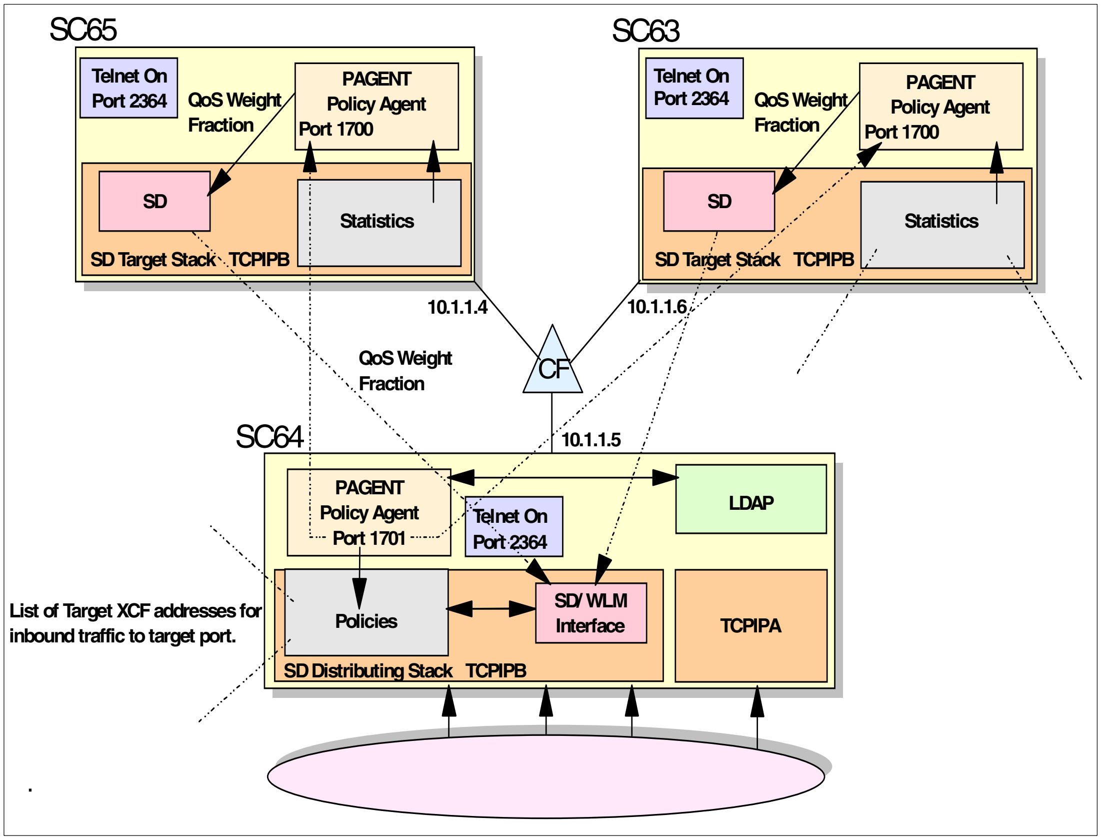


Figure 8-2 PAGENT-to-PAGENT connection

With your current Sysplex Distributor policies already in place, to implement this feature:

1. Add the following entry to the /etc/services directory on all distributing and target stacks in the sysplex.

   ```
   #
   #       Policy Agent QoS Listener and Collector ports
   #
   pagentQosListener  1700/tcp            # Policy Agent Listener thread
   pagentQosCollector 1701/tcp            # Policy Agent Collector thread
   ```

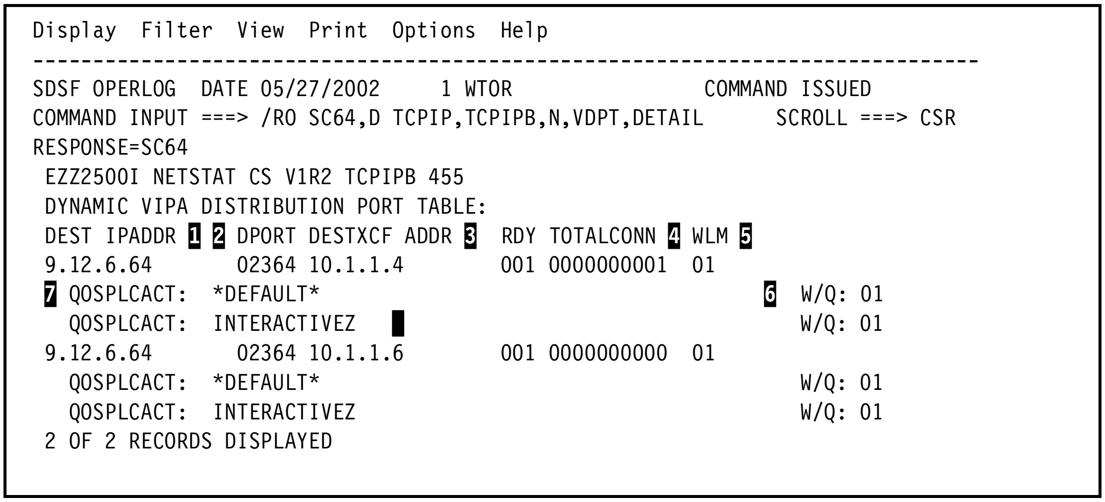
   PAGENT running on a target system will open a listening connection using the pagentQosListener port number. PAGENT running on an distributing system will establish a TCP connection with each PAGENT listener using the pagenQosCollector as the source port.

2. Define the two port numbers mentioned above as reserved ports for PAGENT via the PORT statement in your TCPIP.PROFILE.

Note: It is the first connection attempt for a distributed IP address that causes the PAGENT-to-PAGENT conceitedness to be established. Prior to this all you will see in a connection display will be the PAGENT listen.

8.2.3 Monitoring the Sysplex Distributor QoS

You can use the NETSTAT command to display Sysplex Distributor information. Figure 8-3 shows the display from the distributing stack, SC64, in our sysplex.



```
Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF OPERLOG  DATE 05/27/2002     1 WTOR               COMMAND ISSUED
COMMAND INPUT ===> /RO SC64,D TCPIP,TCPIPB,N,VDPT,DETAIL      SCROLL ===> CSR
RESPONSE=SC64
 EZZ2500I NETSTAT CS V1R2 TCPIPB 455
 DYNAMIC VIPA DISTRIBUTION PORT TABLE:
 DEST IPADDR 1 2 DPORT DESTXCF ADDR 3  RDY TOTALCONN 4 WLM 5
 9.12.6.64       02364 10.1.1.4       001 0000000001  01
 7 QOSPLCACT:  *DEFAULT*                              6  W/Q: 01
   QOSPLCACT:  INTERACTIVEZ   █                          W/Q: 01
 9.12.6.64       02364 10.1.1.6       001 0000000000  01
   QOSPLCACT:  *DEFAULT*                                 W/Q: 01
   QOSPLCACT:  INTERACTIVEZ                              W/Q: 01
 2 OF 2 RECORDS DISPLAYED
```

Figure 8-3 Sysplex Distributor VPDT detail display

1 DEST IPADDR is the distributing VIPA address of our sysplex complex.

2 DPORT is the port number to distribute workload in the sysplex.

3 DESTXCF ADDR is the IP address of the SD target stacks. This address is configured for the dynamic XCF link in the IPCONFIG statement.

4 TOTALCONN is the counter of connections distributed since TCP/IP was started.

5 WLM is the normalized Workload Manager weight passed from Sysplex Distributor.

6 W/Q is the Workload Manager weight value for the target TCP/IP stack after modification by the QoS fraction received from Policy Agent.

7 QOSPLCACT is a QoS policy action name configured in Policy Agent. The `*Default*` entry indicates the WLM and W/Q values used when there is no QosPolicyAction that applies to an incoming connection.

As you can see, we have a single distributed Telnet connection to port, which was distributed to SC65 (10.1.1.4).

To verify that the Sysplex Distributor policy has been successfully enabled, we can check the active policies using the **`pasearch`** command; the command requires a superuser authority. Figure 8-4 on page 179 shows the **`pasearch`** policies.

```
MVS TCP/IP pasearch CS V1R2               TCP/IP Image:      TCPIPB
  Date:                 05/27/2002        Time:  16:47:19

policyRule:             telnetd
  Version:              2  1              Status:            Active   2
  Weight:               2                 ForLoadDist:       TRUE
  Priority:             0                 Sequence Actions:  Don't Care
  ConditionListType:    DNF               No. Policy Action: 1
  policyAction:         interactivez
   ActionType:          QOS               Action Sequence:   0
  Time Periods:
   Day of Month Mask:   1111111111111111111111111111111
   Month of Year Mask:  111111111111
   Day of Week Mask:    1111111  (Sunday - Saturday)
   Start Date Time:     None
   End Date Time:       None
   From TimeOfDay:      00:00             To TimeOfDay:      24:00
   From TimeOfDay UTC:  04:00             To TimeOfDay UTC:  04:00
   TimeZone:            Local
  Net Condition Summary:                  NegativeIndicator: OFF
   RouteCondition:
    InInterface:        0.0.0.0           OutInterface:      0.0.0.0
   HostCondition:
    SourceIpFrom:       0.0.0.0           SourceIpTo:        0.0.0.0
    DestIpFrom:         0.0.0.0           DestIpTo:          0.0.0.0
   ApplicationCondition:
    ProtocolNumFrom:    0                 ProtocolNumTo:     0
    SourcePortFrom:     0                 SourcePortTo:      0
    DestPortFrom:       2364              DestPortTo:        2364
    ApplicationName:                      ApplicationPriority: 0
    ApplicationData:
    ApplicationData:

  Qos Action:           interactivez
    Version:            2  1              Status:            Active    2
    Scope:              DataTraffic   6   OutgoingTOS:       00000000
    Permission:         Allowed
    MaxRate:            0                 MinRate:           0
    MaxDelay:           0                 MaxConn:           0
    Routing Interfaces: 3  3
      Interface Number: 1                 Interface:         10.1.1.4    4
      Interface Number: 2                 Interface:         10.1.1.6    4
      Interface Number: 3                 Interface:         0.0.0.0     5
    RSVP Attributes
     ServiceType:       0                 MaxRatePerFlow:    0
     MaxTokBuckPerFlw:  0                 MaxFlows:          0
    DiffServ Attributes
     InProfRate:        0                 InProfPeakRate:    0
     InProfTokBuck:     0                 InProfMaxPackSz:   0
     OutProfXmtTOSByte: 00000000          ExcessTrafficTr:   BestEffort
```

Figure 8-4 Display policies with the pasearch command

The pasearch report shows all attributes of the policy installed, such as the version 1 of this policy and the policy status 2.

The Routing Interfaces attribute 3 indicates whether this policy is a Sysplex Distributor policy. Three interface attributes have been defined for this policy. The value has to be an IP address of the dynamic XCF link that has been defined for the SD target stack 4 or a value 0.0.0.0 5, which indicates that any target system may be considered for this connection. Those dynamic XCF links are used to route the incoming connection requests. The PolicyScope attribute 6 must specify either DataTraffic or Both to define interfaces using this attribute.

8.2.4 Sysplex Distributor policies in the LDAP Server

In this scenario, the SD policies have been stored in the OS/390 LDAP Server, and is retrieved by the Policy Agent running on the SD distributing stack. The policy used is the same one as shown in the configuration file on page 173, except on this occasion we have included a valid time period for this rule and have also included the interface 0.0.0.0. The rule is only to be active between 08.00 and 18.00 on weekdays.

> **Attention:** Coding Sysplex Distributor policies in the format required for them to be stored in an LDAP Server is not a trivial task. We at the ITSO do not recommend such an undertaking. If you require your Sysplex Distributor policies to be stored in an LDAP Server we strongly suggest that you use the zQoS Manager GUI interface which is explained in Chapter 6, “QoS configuration using the zQoS Manager” on page 125, to do this.

In LDAP this policy would look as follows:

```
dn:cn=sysplexPolicyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyGroup
objectclass:ibm-policyGroupLoadDistributionAuxClass
cn:sysplexPolicyRules
ibm-policyGroupName:sysplexPolicyRules
ibm-policyGroupForLoadDistribution:TRUE

dn:cn=interactivez, cn=actionRepository, cn=policyRepository, ou=zqosMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyActionInstance
objectclass:ibm-policyActionAuxClass
objectclass:ibm-serviceCategoriesAuxClass
cn:interactivez
ibm-policyActionName:interactivez
ibm-policyScope:DataTraffic
ibm-OutgoingTOS:00000000
ibm-MaxRate:0
ibm-DiffServInProfileRate:0
ibm-DiffServInProfileTokenBucket:0
ibm-DiffServExcessTrafficTreatment:BestEffort
ibm-DiffServOutProfileTransmittedTOSByte:00000000
ibm-MinRate:0
ibm-MaxDelay:0
ibm-MaxConnections:0
ibm-Permission:Allowed
ibm-interface:1--10.1.1.4 1
ibm-interface:1--10.1.1.6 1
ibm-interface:1--0.0.0.0 1

dn:cn=telnetd, cn=serverRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
```

```
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-networkingPolicyConditionAuxClass
objectclass:ibm-hostConditionAuxClass
objectclass:ibm-applicationConditionAuxClass
cn:telnetd
ibm-policyConditionName:telnetd
ibm-sourcePortRange:2364:2364

dn:cn=telnetd, cn=sysplexServerRepository, cn=policyRepository, ou=zqosMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-networkingPolicyConditionAuxClass
objectclass:ibm-hostConditionAuxClass
objectclass:ibm-applicationConditionAuxClass
cn:telnetd
ibm-policyConditionName:telnetd
ibm-destinationPortRange:2364:2364

dn:cn=dayshift-0, cn=timeRepository, cn=policyRepository, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-policyTimePeriodConditionAuxClass
cn:dayshift-0
ibm-ptpConditionDayOfWeekMask:0111110
ibm-ptpConditionDayOfMonthMask:111111111111111111111111111111100000000000000000000000000000
00
ibm-ptpConditionTimeOfDayMask:080000:180000
ibm-ptpConditionLocalOrUtcTime:1

dn:cn=telnetd, cn=sysplexPolicyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRule
objectclass:ibm-policySubtreesPtrAuxClass
cn:telnetd
ibm-policyRuleName:telnetd
ibm-policyRuleEnabled:1
ibm-policyRuleConditionListType:2
ibm-policyRuleValidityPeriodList:cn=dayshift-0, cn=timeRepository, cn=policyRepository,
ou=zqosMgrPolicies, o=ITSO,c=US
ibm-policyRulePriority:255

dn:cn=actAssoc-0, cn=telnetd, cn=sysplexPolicyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleActionAssociation
objectclass:ibm-policyActionAuxClass
cn:actAssoc-0
ibm-policyActionName:actAssoc-0
ibm-policyActionOrder:0
ibm-policyActionDN:cn=interactivez, cn=actionRepository, cn=policyRepository,
ou=zqosMgrPolicies, o=ITSO,c=US

dn:cn=servAssoc-0-0, cn=telnetd, cn=sysplexPolicyRules, ou=zqosMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:servAssoc-0-0
```

```
ibm-policyConditionName:servAssoc-0-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:1
ibm-policyConditionDN:cn=telnetd, cn=sysplexServerRepository, cn=policyRepository,
ou=zqosMgrPolicies, o=ITSO,c=US
```

The policies are identified as SD policies by the presence of the ibm-Interface attribute 1 in the ibm_policyAction object class. Here we have defined ibm-interface 0.0.0.0, so that when neither SC63 nor SC65 can handle an FTP connection request, it will be forwarded to SC64 if a Telnet server is listening on port 2364 there.

The second-level PAGENT configuration file also has been updated to communicate with the OS/390 LDAP Server as shown below:

```
ReadFromDirectory
{
     LDAP_Server                  9.12.6.63
     LDAP_Port                    3389
     LDAP_DistinguishedName       cn=LDAP
     LDAP_Password                secret
     LDAP_SessionPersistent       Yes
     LDAP_ProtocolVersion         3
     LDAP_SchemaVersion           3
     SearchPolicyBaseDN           o=ITSO,c=US 1
}

PolicyPerfMonitorForSDR Enable 2
{
     SamplingInterval 60
     LossRatioAndWeightFr 20 25
     LossMaxWeightFr 95
     TimeoutRatioAndWeightFr 50 50
     TimeoutMaxWeightFr 100
}

TcpImage TCPIPB FLUSH NOPURGE 1800
```

1 PAGENT will download all objects under this tree. Note that all SD policy-related objects defined belong to this group.

2 Because the SD performance monitor policy is not supported by the LDAP Server, it has to be defined in the PAGENT configuration file if necessary.

Using the **pasearch** OS/390 UNIX command, you will see the SD policies installed into the SD distributing stack.

```
MVS TCP/IP pasearch CS V1R2            TCP/IP Image:      TCPIPB
  Date:                 05/29/2002       Time:  17:48:22

policyRule:             telnetd
  Version:              3                Status:            Active
  Distinguish Name:     cn=telnetd,cn=sysplexPolicyRules,ou=zqosMgrPolicies,o=ITSO,c=US 1
  Weight:               355              ForLoadDist:       TRUE
  Priority:             255              Sequence Actions:  Don't Care
  ConditionListType:    CNF              No. Policy Action: 1
  policyAction:         interactivez
   ActionType:          QOS              Action Sequence:   0
  Time Periods:
   Day of Month Mask:
   First to Last:       1111111111111111111111111111111
   Last to First:       0000000000000000000000000000000
   Month of Year Mask:  111111111111
   Day of Week Mask:    0111110  (Sunday - Saturday)
   Start Date Time:     None
 End Date Time:       None
   From TimeOfDay:      08:00             To TimeOfDay:      18:00
   From TimeOfDay UTC:  12:00             To TimeOfDay UTC:  22:00
   TimeZone:            Local
  Net Condition Summary:                  NegativeIndicator: OFF
   RouteCondition:
    InInterface:        0.0.0.0           OutInterface:      0.0.0.0
   HostCondition:
    SourceIpFrom:       0.0.0.0           SourceIpTo:        0.0.0.0
    DestIpFrom:         0.0.0.0           DestIpTo:          0.0.0.0
   ApplicationCondition:
    ProtocolNumFrom:    0                 ProtocolNumTo:     0
    SourcePortFrom:     0                 SourcePortTo:      0
    DestPortFrom:       2364              DestPortTo:        2364
    ApplicationName:                      ApplicationPriority: 0
    ApplicationData:

  Qos Action:           interactivez
    Version:            3                 Status:            Active
    Distinguish Name:
cn=interactivez,cn=actionRepository,cn=policyRepository,ou=zqosMgrPolicies,o=ITSO,c=US 1
    Scope:              DataTraffic       OutgoingTOS:       10000000
    Permission:         Allowed
    MaxRate:            0                 MinRate:           0
    MaxDelay:           0                 MaxConn:           0
    Routing Interfaces: 3
      Interface Number: 1                 Interface:         10.1.1.6        2
      Interface Number: 2                 Interface:         10.1.1.4        2
      Interface Number: 3                 Interface:         0.0.0.0         2
    RSVP Attributes
     ServiceType:       0                 MaxRatePerFlow:    0
     MaxTokBuckPerFlw:  0                 MaxFlows:          0
    DiffServ Attributes
     InProfRate:        0                 InProfPeakRate:    0
     InProfTokBuck:     0                 InProfMaxPackSz:   0
     OutProfXmtTOSByte: 00000000          ExcessTrafficTr:   BestEffort
```

1 is the he distinguished names defined for the LDAP objects.

2 is the Routing Interfaces attribute, which indicates that this policy is an SD policy, is defined in the same way as the SD policies are defined in the PAGENT configuration file.

You will find further information about the SD policy and SD performance monitor policy in conjunction with an LDAP Server in *z/OS V1R2 Communications Server: IP Configuration Reference,* SC31-8776.

Part 4

Intrusion detection

9

Overview of Intrusion Detection Services (IDS)

The Intrusion Detection Services (IDS) on CS for z/OS V1R2.0 is designed to protect systems from attacks as well as detect patterns of usage that might indicate impending attacks. Many attacks follow a sequence of information gathering, unauthorized access to resources (information, applications, storage), and denial of service. It can be difficult or, at times, impossible to determine the originator of denial of service attacks. However, correlating information-gathering activities with access violation may help identify an intruder before they succeed.

The IDS supported and discussed throughout Part 4, “Intrusion detection” on page 185, can be categorized into three areas:

- Scan detection
- Attack detection
- Traffic Regulation

The managing of the IDS is done through policies. The policy is designed by the network administrator and based on preconceived events. The policy must include factors such as who, what, where, when, why, and how.

- *Who* is allowed to connect to the host?
- *What* applications/ports are clients allowed to use?
- *Where* is the attack/intruder/traffic emanating from?
- *When* should I consider something to be an attack or scan?
- *How* is my system affected by the attack, scan, or traffic?

The IDS policy information resides on an LDAP Server that may or may not be located on the local host. The IDS policy syntax is based on the definition of the elements of policies provided by the LDAP Server product. The definition of the elements of policies is known as the schema, and z/OS CS provides the schema definition for policies in a set of sample files that is to be installed on the LDAP Server.

The policy information is loaded into the Policy Agent (PAGENT) application during PAGENT start-up. All IDS policies allow the logging events to a specified message level in syslogd and/or the system console. Most IDS policies support discarding packets when a specified limit is reached. Most IDS policies support writing statistics records to the INFO message level of syslogd on a specified time interval and/or if exception events have occurred. All IDS policies support tracing all or part of the triggering packet to an IDS-specific CTRACE facility, SYSTCPIS. IDS assigns a correlator value to each event. Messages written to the system console and syslogd and records written to the IDS ctrace facility all use this correlator. A single detected event may involve multiple packets. The correlator value identifies which message and packets are related to each other.

The type of policy written can either be a Scan Policy, Attack Policy, or a Traffic Regulation Policy. The following sections give a detailed description of these policies.

9.1 Scan policies

Scans are recognized as the result of multiple information gathering events from a single source IP within a defined period of time. Scanning, itself, is not necessarily harmful and may be part of normal operation. However, many serious attacks, especially access violation attacks, are preceded by information gathering scans. Because by their nature scans must use reliable source IP addresses, they can be monitored and the data processed to help prevent or determine the origins of a previous attack.

The IDS support defines a scanner as a source host that accesses multiple unique resources (ports or interfaces) over a specified period of time. The number of unique resources (threshold) and the time period (interval) can be specified via policy. Two categories of scans are supported:

- Fast scan

  Many resources rapidly accessed in a short time period (usually less than five minutes, and program driven), as shown in Figure 9-1 on page 190.

- Slow scan

  Different resources intermittently accessed over a longer period of time (many hours). This could be a scanner trying to avoid detection, as shown in Figure 9-2 on page 190.

A fast scan scenario my be one in which an attack is based on the information provided through a program that loops through ports 1–1025 (normally the ports used by the server for listening ports), determining which ports have active listeners. This information may be the basis for a future attack. A slow attack is more deliberate; occasional packets may be sent out to different ports over a long period of time with the same fundamental purpose, obtaining host information.

The same port being accessed will not generate multiple event records, for example, if a client from the same source IP address generates twenty connections to port 23 (TN3270 server). This is not considered a scan because only one unique resource has been accessed.

> **Note:** Scan policies do not provide the ability to reject a connection. The actual rejecting of the connection based on the source IP address must be configured in the Traffic Regulation policy or firewall.

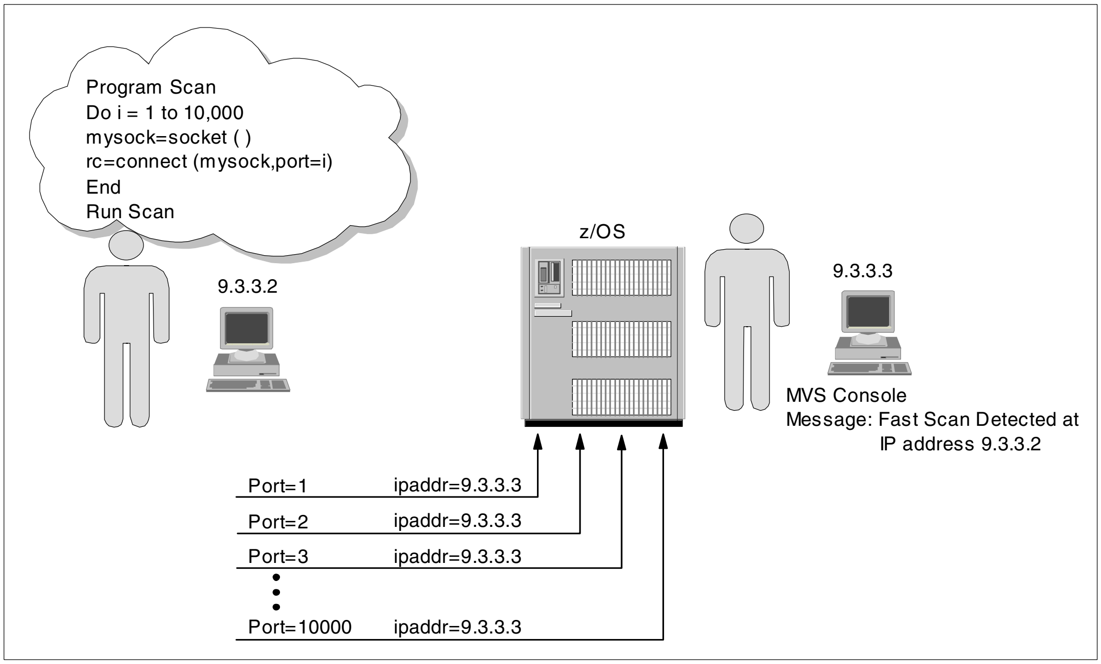


Figure 9-1 Fast scan

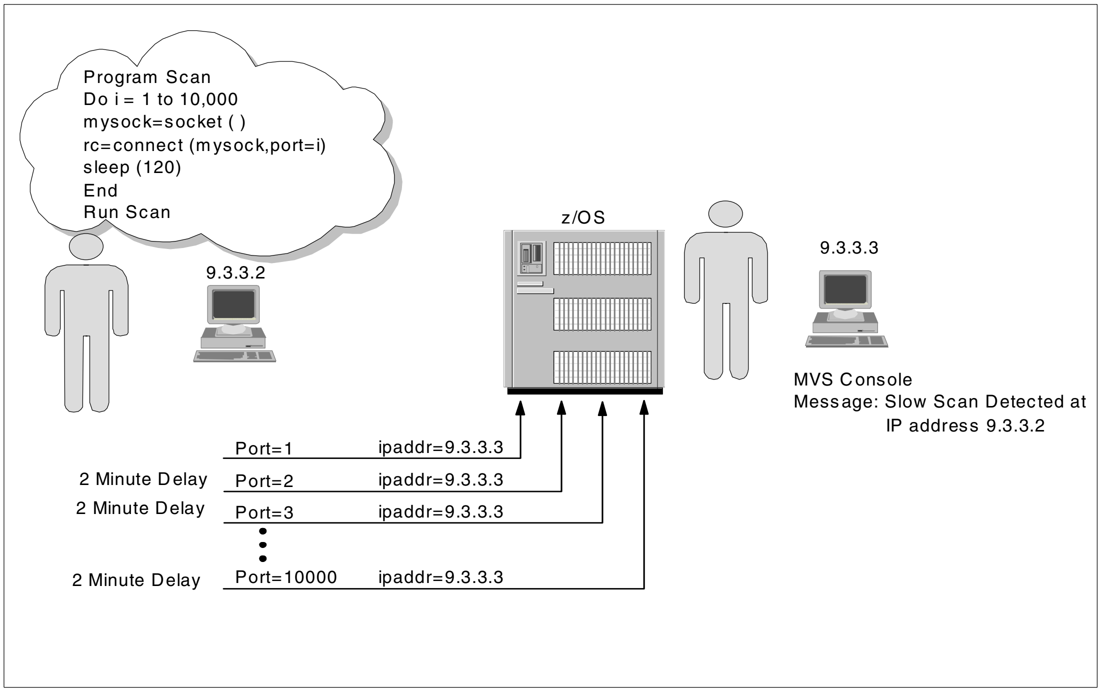


Figure 9-2 Slow scan

9.1.1 Scan policy parameters

Scan policy provides the ability to control the following parameters that define a scan:

- Fast scan time interval.
- Slow scan time interval.
- Fast scan threshold.

- Slow scan threshold.
- Exclude well-known legitimate scanners via an exclusion list.
- Specify a sensitivity level by port or port range (to reduce performance impacts).
- Notify the installation of a detected scan via console message or syslogd message.
- Trace potential scan packets.

The policy allows the user to set a sensitivity level. This is known as policy-specified sensitivity. This is used in parallel with the categorization of the individual packets to determine if a packet should be counted as a scan event. The event classification is either a normal, possibly suspicious, or a very suspicious event. The mapping of sensitivity to event classification is shown in Table 9-1. This logic is used to control the performance impact and analysis load of scan monitoring by only counting those individual packets where the chart indicates a count value. This value is then added with the current count total of scan events and compared with the threshold value to determine if we have met or exceeded the threshold in a specified time interval.

Table 9-1 Sensitivity table

Sensitivity (from policy)	**Normal event**	**Possibly suspicious event**	**Very suspicious event**
Low	—	—	Count
Medium	—	Count	Count
High	Count	Count	Count

9.1.2 Scan events

Scan events are classified into ICMP, UDP port, and TCP port scans categories. The scan categories are described here:

- ICMP scan

  ICMP requests (echo, information, timestamp, and subnet mask) are used to obtain or map network information. The type of ICMP request determines the event classification.

- TCP port scans

  TCP is a stateful protocol. There are many different events that may be classified as normal, possibly suspicious, or highly suspicious.

- UDP port scans

  UDP is stateless. The stack is unable to differentiate between a client port and a server port. A scanner sending messages to many ephemeral ports looks very similar to a DNS server sending replies to many clients on ephemeral ports. TCP/IP configuration allows UDP ports to be RESERVED, therefore, restricting a port so that it cannot be used.

Table 9-2 describes how the TCP/IP stack places packets into an event classification based on the request type (ICMP) and Table 9-3 on page 192 on the socket state (TCP and UDP).

Table 9-2 ICMP event classification

Request type	**Destination address**	**Event classification**
Any	Subnet base or broadcast	Very suspicious
Information or subnet mask	Single host	Possibly suspicious
Echo or timestamp	Single host	Possibly suspicious

Request type	Destination address	Event classification
Echo with IP option record route or record timestamp	Single host	Normal

Table 9-3 TCP/UPD event classification

Socket state	Event	Event classification
UDP	—	—
Restricted (RESERVED)	Receive any packet	Very suspicious
Unbound, not restricted	Receive any packet	Possibly suspicious
TCP	—	—
Any state	Receive unexpected flags	Very suspicious
Restricted (RESERVED)	Receive any packet	Very suspicious
Unbound, not restricted	Receive any packet	Possibly suspicious
Listen	Receive stand-alone SYN	No event
Half open connection	Receive ACK	Normal
Half-open connection	Receive RST	Possibly suspicious
Half-open connection	Final time out	Very suspicious
Any connected state	Seq# out of widow	Normal
Any connected state	Receive stand-alone SYN	Normal
Any connected state	Final time-out	Possibly suspicious

Any countable scan event will count against an origin source IP address. The total number of countable events from all categories is compared to the policy thresholds. When an origin source IP address has exceeded the policy-defined fast or slow threshold an event may be sent to the TRMD for logging to syslogd, a console may be issued, and optionally a packet trace record issued. This is all dependant upon the notification actions set in the action of the policy. Once a scan event is logged for a particular source IP address, no further scan events will be reportable within the specified fast interval. The intervals and thresholds for fast and slow scan are global. Only one definition of them is allowed across all event categories at a given point in time.

9.1.3 False positive scans

IDS attempts to reduce the recording of false scan events. This can be manually coded in the policy by excluding a source IP address, port, or subnet. This is useful if you have a particular client that probes the TCP/IP stack for general statistical information. Also, only unique events from a source IP address are counted as a scan event. An event is considered unique if the four-tuple, client IP address, client port, server IP address, and server port are unique as well as the IP protocol for this scan interval. In the case of ICMP, a packet is unique if the type has not been seen before within this scan interval.

9.2 Attack policies

An *attack* is defined as an assault on system security that derives from an intelligent threat. It is an intelligent act that is a deliberate attempt to evade security services and violate the security policy of a system. An attack may in the form of a single packet or multiple packets. There are two types of attacks, active and passive. For more information on passive and active attacks, reference RFC 2828.

- An *active attack* is designed to alter system resources or affect their operation.
- A *passive attack* is designed to learn or make use of system information but not affect system performance.

The attack policies designed for IDS are based on active attacks. One may consider scanning to be a more passive in nature attack.

IDS attack policy allows the network administrator to provide network detection for one or more categories of attacks independently of each other. In general, the types of actions that can be specified for an attack policy are notifications (that is, event logging, statistics gathering, packet tracing) and discarding of the attack packets.

9.2.1 IDS attack categories

The IDS categories of attacks are described in Table 9-4.

Table 9-4 Attack categories

Category	Attack description	Actions
Malformed packets	There are numerous attacks designed to crash a system's protocol stack by providing incorrect partial header information. The source IP address is rarely reliable for this type of attack.	TCP/IP stack: Always discards malformed packets. IDS policy: May provide notification.
Inbound fragment restrictions	Many attacks are the result of fragment overlays in the IP or transport header. This support allows you to protect your system against future attacks by detecting fragmentation in the first 256 bytes of a packet.	TCP/IP stack: No default action. IDS policy: May provide notification and cause the packet to be discarded.
IP protocol restrictions	There are 256 valid IP protocols. Only a few are in common usage today. This support allows you to protect your system against future attacks by prohibiting those protocols that you are not actively supporting.	TCP/IP stack: No default action. IDS policy: May provide notification and cause the packet to be discarded.
IP option restriction	There are 256 valid IP options, with only a small number currently in use. This support allows you to prevent misuse of options you are not intentionally using. Checking for restricted IP options is performed on all inbound packets, even those forwarded to another system.	TCP/IP stack: No default action. IDS policy: May provide notification and cause the packet to be discarded.

Category	Attack description	Actions
UDP perpetual echo	Some UDP applications unconditionally respond to every datagram received. In some cases, such as Echo, CharGen or TimeOfDay, this is a useful network management or network diagnosis tool. In other cases it may be polite application behavior to send error messages in response to incorrectly formed requests. If a datagram is inserted into the network with one of these applications as the destination and another of these applications spoofed as the source, the two applications will respond to each other continually. Each inserted datagram will result in another perpetual echo conversation between them. This support allows you to define the application ports that exhibit this behavior.	TCP/IP stack: No default action. IDS policy: May provide notification and cause packet to be discarded.
ICMP redirect restrictions	ICMP redirect packets can be used to modify your routing tables.	TCP/IP stack: Will discard ICMP redirects if IGNOREREDIRECT is coded in the tcpip.profile. IDS policy: May provide notification and disable redirects (this can optionally be coded as a parameter in the tcpip.profile).
Outbound raw restrictions	Most network attacks require the ability to craft packets that would not normally be built by a proper protocol stack implementation. This support allows you to detect and prevent many of these crafting attempts so that your system is not used as the source of attacks. As part of this checking, you can restrict the IP protocols allowed in an outbound RAW packet. It is recommended that you restrict the TCP protocol on the outbound raw rule.	TCP/IP stack: No default action. IDS policy: May provide notification and cause the packet to be discarded.
TCP SYNflood	One popular denial of service attack is to flood a public server with connection requests from invalid or nonexistent source IP addresses. The intent is to use up the available slots for connection requests and thereby deny legitimate access from completing.	TCP/IP stack: Provides internal protection against SYN attack. IDS policy: May provide notification.

9.2.2 Attack policy notification

The IDS attack policy (object class name ibm-idsNotification) notification allows attack events to be logged to syslogd and/or the system console. For all attack categories except flood, a single packet triggers an event. To prevent message flooding to the system console, you can specify the maximum number of console messages to be logged per attack category within a five-minute interval (ibm-idsMaxEventMessage). There is no default so it is recommended that you code a maximum number of event messages that are to be written to the console. To prevent message flooding to syslogd, a maximum of 100 event messages per attack category will be logged to syslogd within a five-minute interval.

Note: The console messages provide a subset of the information provided in the syslogd messages.

9.2.3 Attack policy statistics

The IDS attack policy statistics action provides a count of the number of attack events detected during the statistics interval. The count of attacks is kept separately for each category of attack (for example, malformed) and a separate statistics record is generated for each. If you want to turn on statistics for attacks, it is recommended that you specify exception statistics (ibm-idsTypeActions:EXCEPTSTATS). With exception statistics, a statistics record will only be generated for the category of attack if the count of attacks is non-zero. If statistics are requested (ibm-idsTypeActions:STATISTICS) a record will be generated every statistics interval regardless of whether an attack has been detected during that interval.

9.2.4 Attack policy tracing

The IDS attack policy tracing uses the component trace facility SYSTCPIS. The attack policy tracing attributes are ibm-idsTraceData and ibm-idsTraceRecordSize, which indicate if packets associated with the attack events are to be traced. For all attack categories except flood, a single packet triggers an event and the packet is traced. In the case of a flood, a maximum of 100 attack packets per attack category will be traced during a five-minute interval.

Note: In order to use the attack policy tracing via the ctrace component SYSTCPIS, the component must be started. Reference *z/OS Version 1 Release 4 Communications Server: IP Diagnosis Guide*, GC31-8782, for more information.

9.3 Traffic Regulation (TR) policies

The IDS TR policies are used to limit memory resource consumption and queue delay time during peak loads. There are two types of TR policies that will be discussed in the subsequent sections, TCP and UDP Traffic Regulation policies.

9.3.1 TR TCP policy information

The IDS TR policies for TCP ports limit the total number of connections an application has active at one time. This can be used to limit the number of address spaces created by forking applications such as otelnetd. The TR TCP terminology is very important when coding the policy to ensure the desired goal is achieved. The following section describes the TR TCP terminology.

Connections

Connections can be separated into two groups. Total connections and number of available connections.

Total connections

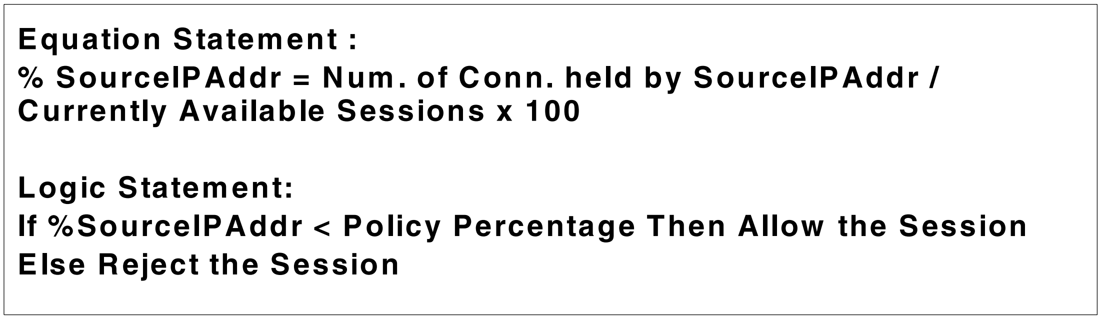
This is the total number of connections that are coded in your policy. This number can never be exceeded for a particular port.

Number of available connections

This is the total number of available connections, which is equal to the sessions in use subtracted from the total connections. This value is used in the fair share algorithm.

Fair share algorithm

The fair share algorithm is designed to limit the number of connections available to any source IP address. The algorithm is based on the percentage of the available remaining connections for a particular port compared with the total connections already held by the source IP address for that port. The fair share equation and logic statements are shown in Figure 9-3.



Equation Statement :
% SourceIPAddr = Num. of Conn. held by SourceIPAddr /
Currently Available Sessions x 100

Logic Statement:
If %SourceIPAddr < Policy Percentage Then Allow the Session
Else Reject the Session

Figure 9-3 Fair share logic

Note: If a host does not currently have any connections open on the port and connections are available, a host will always be allowed at least one connection.

QoS policy

Multi-user source IP addresses may be allowed a larger number of connections by specifying a QoS policy with a higher number of connections (MaxConnections) than allowed by the TR policy. TR will honor the QoS Differentiated Services Policy if the port is *not* in a *constrained state.* A QoS exception is made only when QoS Differentiated Services Policy is applied for the specific source server port and specific outbound client destination IP address.

Constrained state

TR TCP generates a constrained event when a port reaches approximately 90 percent of its connection limit (total connections). An unconstrained event is generated when the port falls below approximately 88 percent of its limit. This two percent deviance is designed to avoid message flooding.

9.3.2 TR UDP policy information

Traffic Regulation for UDP connections can be done in two ways, either through the UDPQUEUELIMIT parameter in the TCPIP.PROFILE or by coding a TR UDP policy. If both are in affect, the TR UDP policy takes priority.

UDPQUEUELIMIT

Traffic Regulation for UDP-based applications can be provided through the TCPIP.PROFILE statement of UDPQUELIMIT. This statement relates to inbound packets for bound UDP ports. Packets are queued until the queue limit is reached or buffer memory is exhausted. If NOUDPQUEUELIMIT is coded, any single bound port under a flood attack or with a stalled application could consume all available buffer storage. It is recommended that

UDPQUEUELIMIT always be set to active. This limits the amount of storage that can be consumed by inbound datagrams for any single bound port. Sockets that use the Pascal API have a limit of 160 KB in any number of datagrams. Sockets that use other APIs have a limit of 2000 datagrams or 2880 KB.

> **Note:** If a policy is in effect for a UDP port, the queue limit size is controlled by the policy for that port.

TR UDP policies

IDS TR policies for UDP ports specify one of four abstract queue sizes for specified bound IP addresses and ports. The four abstract sizes are VERY_SHORT, SHORT, LONG, and VERY_LONG. The abstract size is comprised of two values, the number of packets and the total number of bytes on the queue. If either one of these values is exceeded, inbound data is discarded. See Table 9-5 for the internal values.

Table 9-5 TR UDP abstract queue information

Abstract size	**Number of packets**	**Queue limit**
VERY_SHORT	16	32 KB
SHORT	256	512 KB
LONG	2048	4 MB
VERY_LONG	8192	16 MB

Most UDP applications have time-out values based on human perceptions of responsiveness. These values tend to stay constant while system processing speeds and network delivery speeds continue to advance rapidly. This may require the physical sizes of these queues to change over time. For performance reasons, sockets that use the Pascal API will only enforce the byte limit. Sockets that use other APIs will enforce both limits. Sockets without a policy specified for their port can use the existing UDPQUEUELIMIT mechanism.

For applications that can process datagrams at a rate faster than the average arrival rate, the queue acts as a speed matching buffer that shifts temporary peak workloads into following valleys. The more that the application processing rate exceeds the average arrival rate and the larger the queue, the greater the variation in arrival rates that can be absorbed without losing work. Very fast applications with very bursty traffic patterns may benefit from LONG or VERY_LONG queue sizes.

For applications that consistently receive datagrams at a higher rate than they are able to process them, the queue acts to limit the effective arrival rate to the processing rate by discarding excess datagrams. In this case the queue size only influences the average wait time of datagrams in the queue and not the percentage of work lost. In fact, if the wait time gets too large, the peer application may have given up or retransmitted the datagram before it is processed. Slow applications with consistently high traffic rates may benefit from SHORT queue sizes. In general, client-side applications will tend to have lower system priority, giving them lower datagram processing rates. They also tend to have much lower datagram arrival rates. Giving them SHORT or VERY_SHORT queue sizes may reduce the risk to system buffer storage under random port flood attacks with little impact on percentage of datagrams lost.

10

IDS configuration using zIDS Manager

This chapter serves as a pseudo user's guide for the zIDS Manager tool. The zIDS Manager is a tool designed to allow a network administrator to produce the following:

- A file the LDAP Server can process (using either LDAP Version 2 or 3)
- A configuration file for the Policy Agent (PAGENT)

This chapter contains the following sections:

10.1 What a zIDS is

zIDS Manager enables centralized configuration of Intrusion Detection policy for z/OS V1R2 using LDAP as a policy repository. It provides a user-friendly interface with help panels to free network administrators from having to know LDAP Policy Schema and the complexity of directly writing to an LDAP Server.

The zIDS Manager is a tool designed to allow a network administrator to produce the following:

- A file the LDAP Server can process (using either LDAP version 2 or 3)
- A configuration file for the Policy Agent (PAGENT)

zIDS Manger's Graphical User Interface (GUI) provides a user-friendly front end for the entry of policy information. There is also the flexibilty to save the (possibly incomplete) information you entered in an XML file format on your local machine for future use. Once the policies are complete, the tool can covert the XML file to an LDIF file and send the information to the LDAP Server and/or optionally save the file locally in the LDIF file format.

zIDS Manager also produces a configuration file with the LDAP Server information required by PAGENT. This file can be sent via FTP or moved to and placed in the PAGENT configuration file manually.

The IDS functionality provided by PAGENT must use the LDAP Server to retrieve its policy information. The LDAP Server requires that the policies be coded in accordance with RFC 2849 and as per the RFC, transferred to the LDAP Server in an LDIF file format. IDS policies cannot be configured in the PAGENT configuration file. Based on this premise, there are two methods for generating the LDIF file:

- Manually coding the policies (LDIF) file, and transferring the information to the LDAP Server
- Using the zIDS Manager to generate the LDIF file and using it to automatically send the information to the LDAP Server

We at the ITSO recommend using the zIDS Manager tool to generate and transfer the policy information. The primary reason is that the network administrator does not need to learn LDAP syntax in order to create a policy. Also, the amount of time saved by using the zIDS Manager to generate the LDIF file as opposed to manually coding the LDIF file is significant.

Figure 10-1 on page 201 shows the communication flow between the zIDS Manager, the LDAP Server, and the PAGENT application. This chapter focuses on the first flow, the building of the policies and transfer of the policies to the LDAP Server.

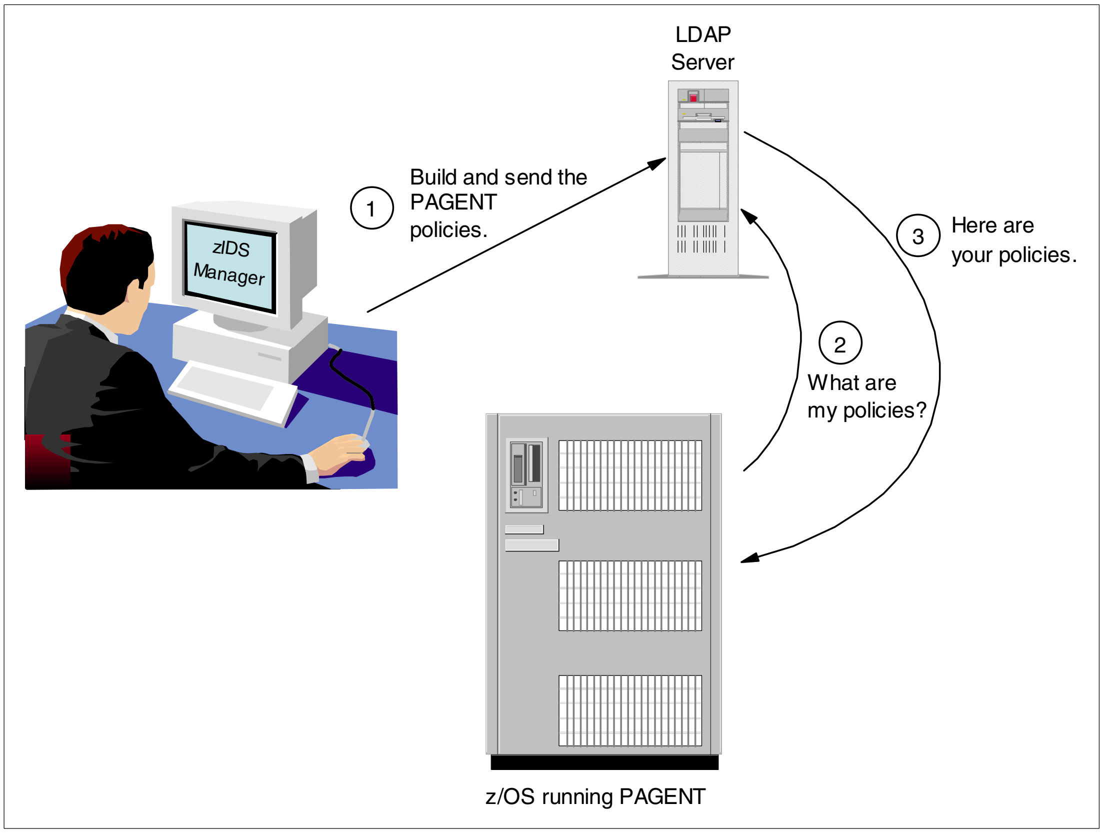


Figure 10-1 Policy flow

zIDS Manager is a tool for network administrators. Therefore, before you begin you should:

- Read the chapter on policy-based networking in *z/OS V1R2 Communications Server: IP Configuration Guide,* SC31-8775.
- Have information about the LDAP Server to be used, that is, the server address and port number, the LDAP protocol version (2 or 3), whether a backup LDAP Server is used, and whether SSL is used.
- Be familiar with your particular environment so that you can make decisions on what events are to be detected under what circumstances and what action to take.

10.2 Requirements and support

This section outlines the requirements and support of the zIDS GUI.

10.2.1 Requirements

The IDS Manager requires Java 1.3.1 and Windows 2000 or Linux to run. The Java executable can be obtained at the following URL:

`http://java.sun.com/`

10.2.2 Support - Legal notice

IBM provides this code on an *as is* basis without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability or fitness for a particular purpose.

Support is provided on a “best effort” basis through a newsgroup. Visit the newsgroup ibm.software.commserver.os390.zids-manager on the server news.software.ibm.com.

The tested operating systems have been Linux and Windows 2000. Any operating system that can support Java 1.3.1 should be able to run the application.

10.3 Download and installation

The download and installation instructions are written for Windows 2000 and Linux. The following information and executables are located at:

`http://www-3.ibm.com/software/network/commserver/downloads/zidsmanager.html`

10.3.1 Windows 2000 steps

The steps for Windows 2000 are:

1. Download this file to your Windows system: zIDSManager.zip.
2. Use an unzip program to extract zIDSManager.exe.
3. Execute zIDSManager.exe.
4. Go to **Start -> Programs-zIDS Manager**.

10.3.2 Linux steps

The steps for Linux are:

1. Download this file to your Linux system: zidsmgr.tar.
2. Untar the file with tar -xvf zidsmgr.tar.
3. Execute ./zidsmgr.

10.4 Using the GUI

This section is intended to help the network administrator manage and understand the Graphical User Interface provided. Each first level directory will be discussed and screen captures provided to assist in the education. The first level directories are:

- zIDS Manager configuration (10.4.1, “zIDS Manager configuration” on page 203)
- PAGENT configuration (10.4.2, “PAGENT configuration” on page 204)
- Work with IDS objects/rules (10.4.3, “Work with IDS objects/rules” on page 208)

Upon completion of this chapter, the reader will have created

- Reusable objects
- A scan global policy
- An attack, scan event, and TR TCP (condition, action, and policy)

The first window displayed when starting the zIDS Manager is shown in Figure 10-2 on page 203.

Note: zIDS Manager help is available via the **Help** menu option. If detailed information is needed for a particular field, place the cursor in the desired field and press the F1 key.

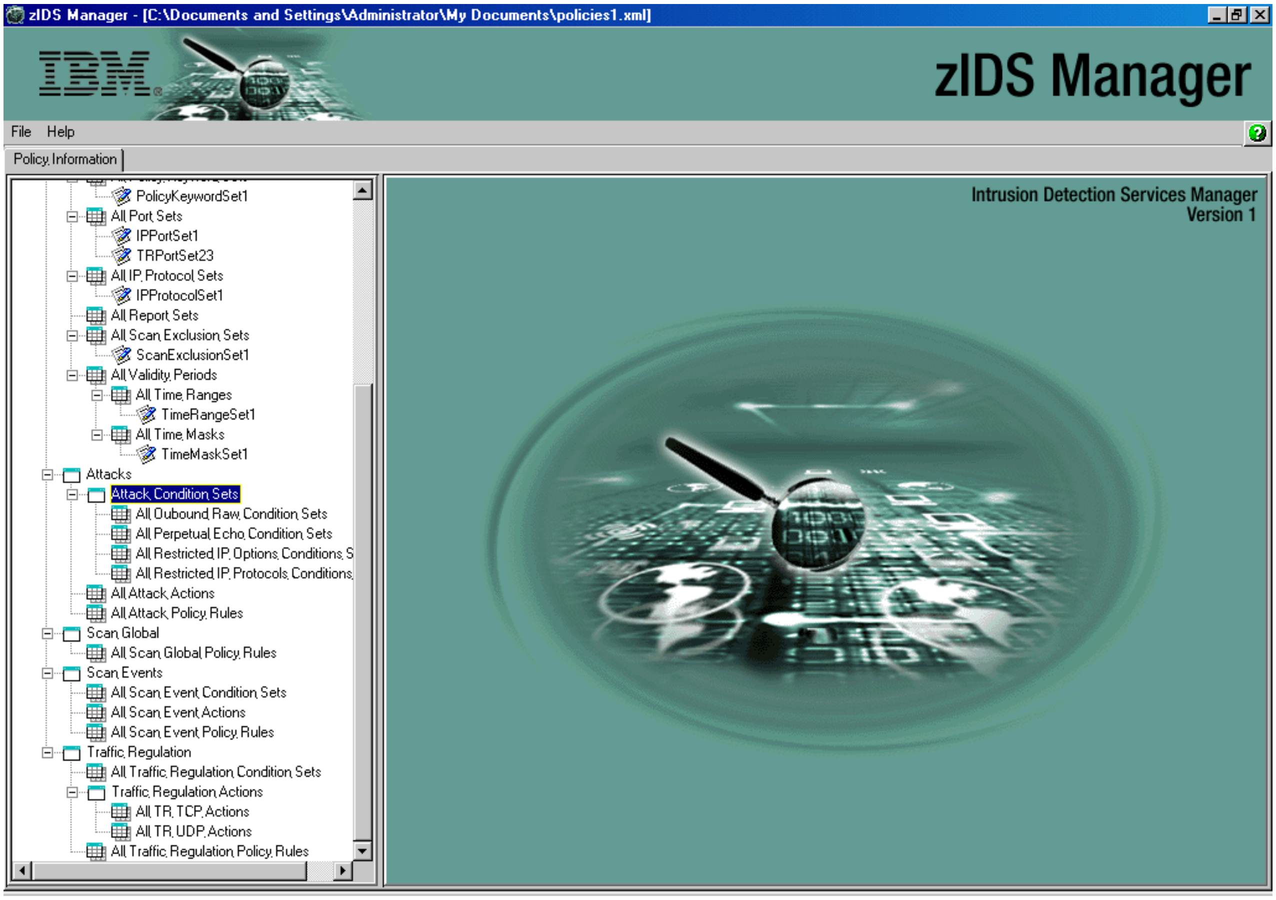
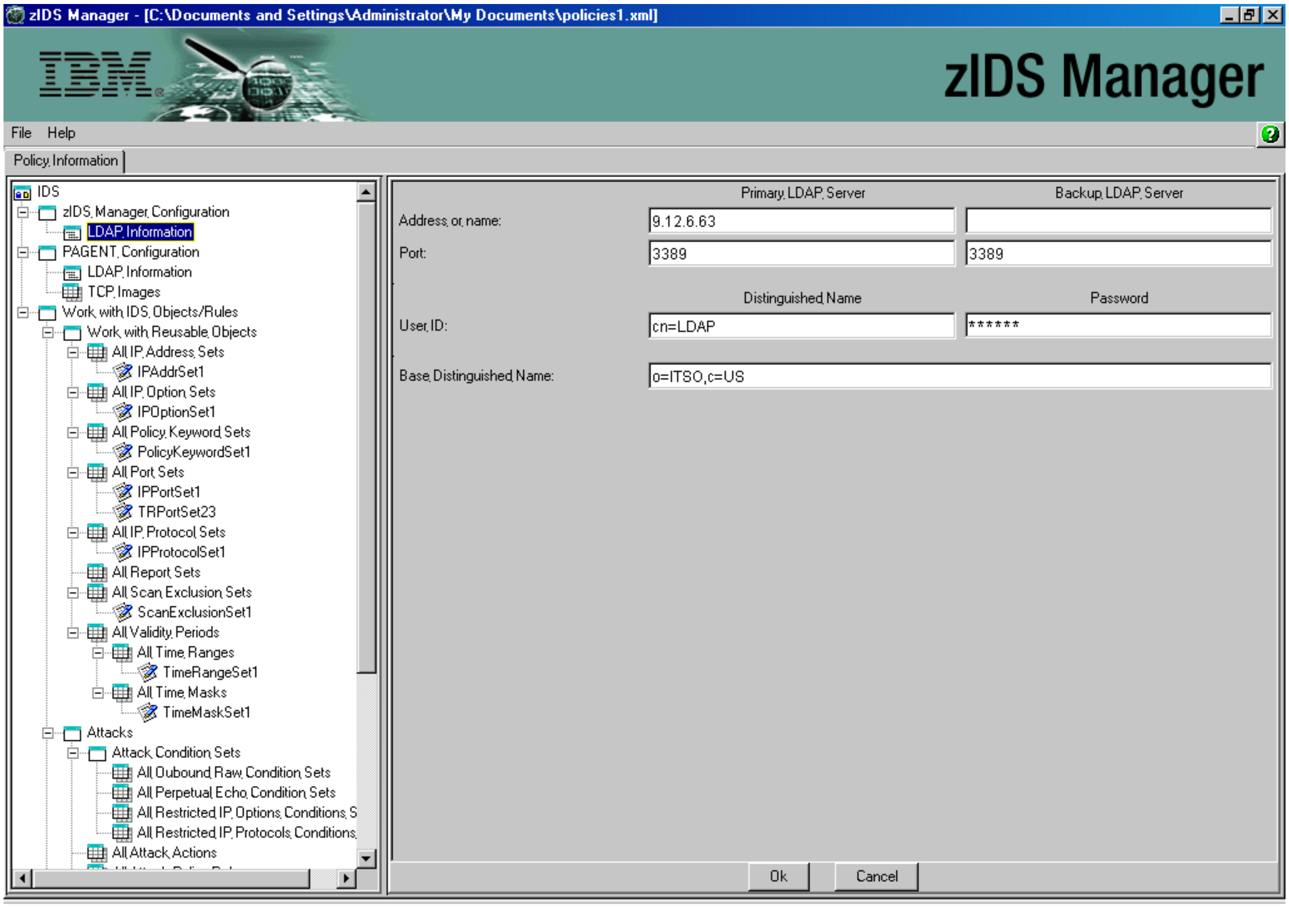

zIDS Manager - [C:\Documents and Settings\Administrator\My Documents\policies1.xml]
IBM
zIDS Manager
File
Help
Policy Information
PolicyKeywordSet1
All Port Sets
IPPortSet1
TRPortSet23
All IP Protocol Sets
IPProtocolSet1
All Report Sets
All Scan Exclusion Sets
ScanExclusionSet1
All Validity Periods
All Time Ranges
TimeRangeSet1
All Time Masks
TimeMaskSet1
Attacks
Attack Condition Sets
All Oubound Raw Condition Sets
All Perpetual Echo Condition Sets
All Restricted IP Options Conditions S
All Restricted IP Protocols Conditions
All Attack Actions
All Attack Policy Rules
Scan Global
All Scan Global Policy Rules
Scan Events
All Scan Event Condition Sets
All Scan Event Actions
All Scan Event Policy Rules
Traffic Regulation
All Traffic Regulation Condition Sets
Traffic Regulation Actions
All TR TCP Actions
All TR UDP Actions
All Traffic Regulation Policy Rules
Intrusion Detection Services Manager
Version 1

Figure 10-2 zIDS Manager

10.4.1 zIDS Manager configuration

One of the first steps in using zIDS Manger is to configure the LDAP Server settings. This is done from the section zIDS Manager Configuration by selecting **LDAP Information**. The settings we used are shown in Figure 10-3 on page 204. This sets up the configuration information that is needed for communication between the zIDS Manager and the LDAP Server.



	Primary LDAP Server	Backup LDAP Server
Address or name:	9.12.6.63	
Port:	3389	3389
	Distinguished Name	Password
User ID:	cn=LDAP	******
Base Distinguished Name:	o=ITSO,c=US	

Figure 10-3 zIDS LDAP configuration information

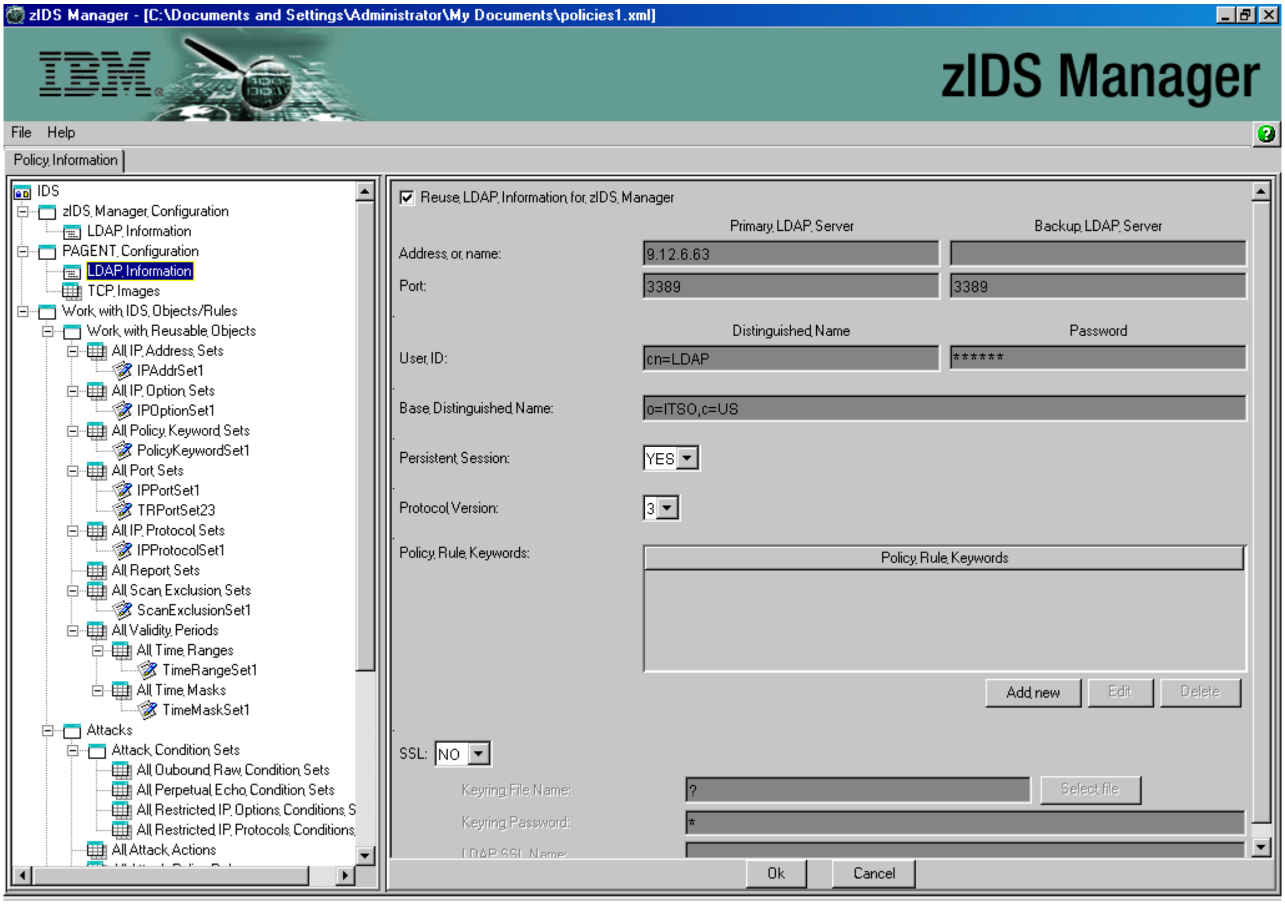
> **Note:** There is not an option for SSL. The connection between the zIDS Manager and the LDAP Server is a non-secure connection.

zIDS Manager to LDAP Server communication

Verify that you can communicate with the LDAP Server by clicking **File -> Send to LDAP**. This is considered successful if you receive the Creating LDIF icon with the corresponding text `Updating LDAP. Please wait` for a few seconds and are then returned control with the icon disappearing and no error messages. Keep in mind that we are not sending any new policy information to the LDAP Server; we basically sent the default IBM-provided policy information for connectivity verification. This same step must be repeated after the policies have been coded and saved to a file by selecting **File -> Save** or **File -> Save As** from the menu.

10.4.2 PAGENT configuration

Next we will configure the PAGENT configuration information by selecting **PAGENT Configuration**. Now select **LDAP Information** and the screen appears as shown in Figure 10-4 on page 205. We will use the same information as entered for the LDAP Information in the zIDS Manger Configuration section.



☑ Reuse LDAP Information for zIDS Manager

	Primary LDAP Server	Backup LDAP Server
Address or name:	9.12.6.63	
Port:	3389	3389

	Distinguished Name	Password
User ID:	cn=LDAP	******

Base Distinguished Name: o=ITSO,c=US

Persistent Session: YES

Protocol Version: 3

Policy Rule Keywords:

SSL: NO

Keyring File Name: ?

Keyring Password: *

LDAP SSL Name:

Figure 10-4 PAGENT LDAP configuration information

We can now configure the TCP images. This will specify what TCP/IP stacks are supporting the policies. First, click **TCP Images**. You will see a summary of information in the right pane. If you right-click you will see the window shown in Figure 10-5 on page 206. This allows you to add multiple TCP/IP images to the configuration file.

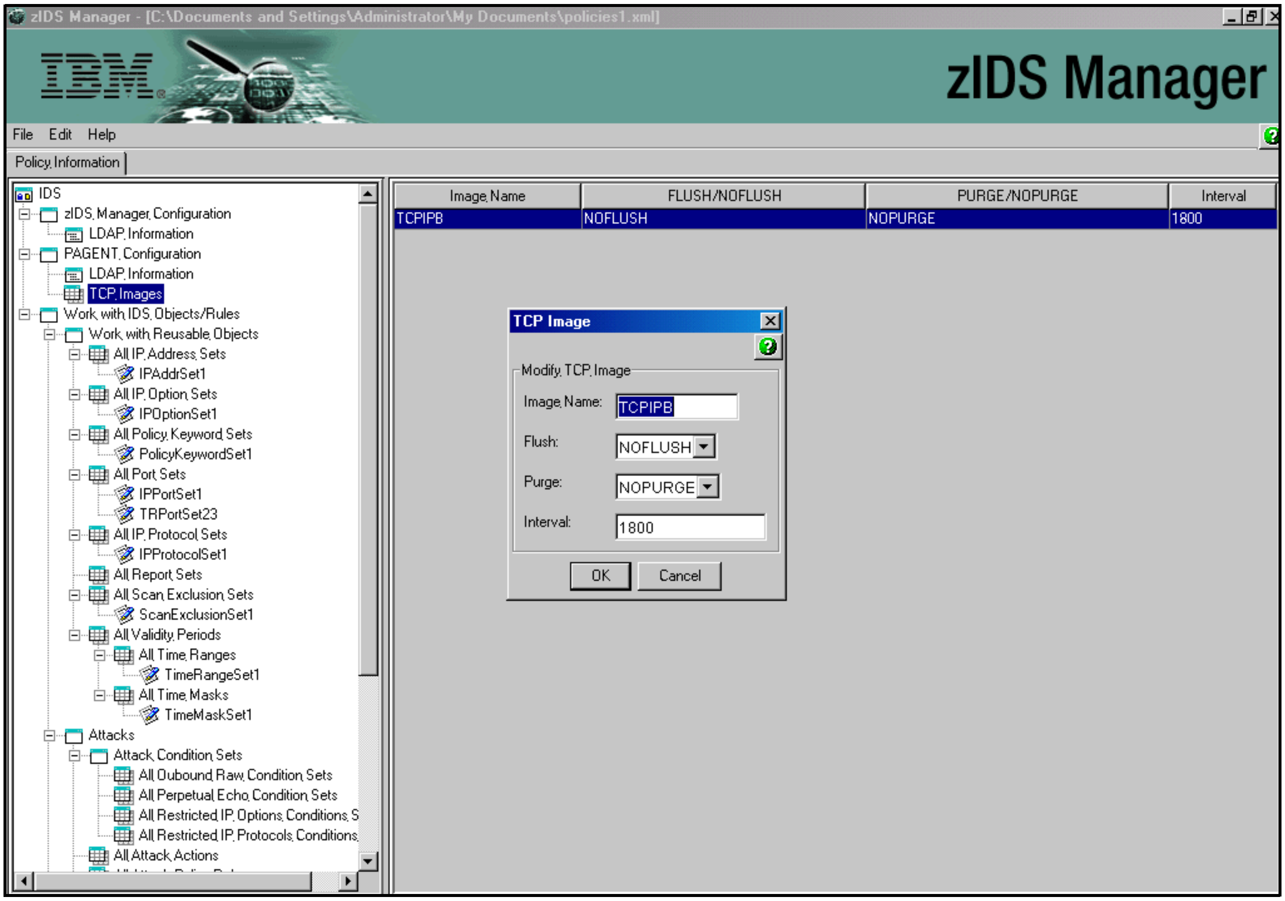


Image Name	FLUSH/NOFLUSH	PURGE/NOPURGE	Interval
TCPIPB	NOFLUSH	NOPURGE	1800

Image Name: TCPIPB

Flush: NOFLUSH

Purge: NOPURGE

Interval: 1800

Figure 10-5 PAGENT LDAP TCP/IP configuration

Once your TCP/IP image has been created you have the ability to add, modify, or delete the TCP/IP Image object by highlighting it in the right pane right-clicking the object. The pop-up menu that appears is shown in Figure 10-6 on page 207.

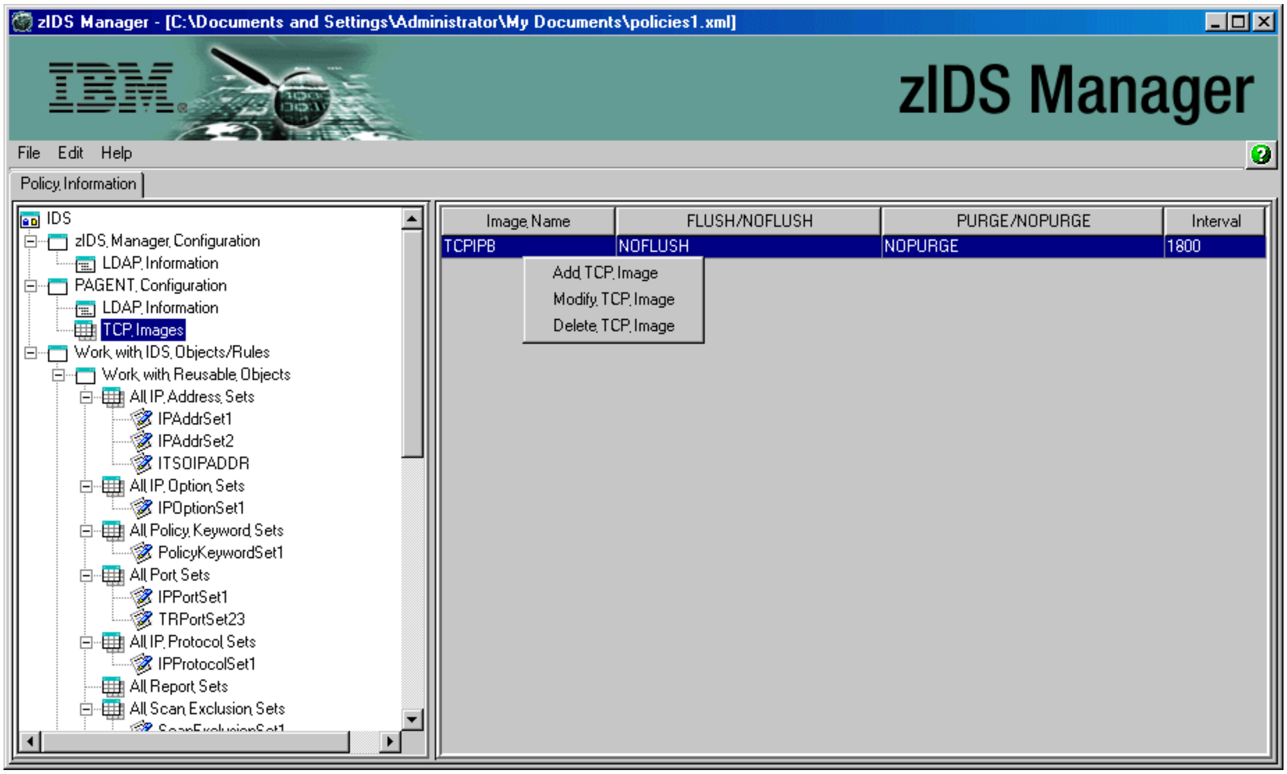


Image Name	FLUSH/NOFLUSH	PURGE/NOPURGE	Interval
TCPIPB	NOFLUSH	NOPURGE	1800

Figure 10-6 Add, modify, or delete TCP/IP image object

At this point we are ready to generate the PAGENT configuration file.

PAGENT configuration file

The PAGENT configuration information must be saved to a text file after providing the necessary information, which we did above. The file is save by selecting **File -> Save As** and then choosing the .conf format, which represents PAGENT LDAP configuration file. An example of the text output is displayed in Figure 10-7.

```
ReadFromDirectory
{
     LDAP_Server                  9.12.6.63
     LDAP_Port                    3389
     LDAP_DistinguishedName       cn=LDAP
     LDAP_Password                secret
     LDAP_SessionPersistent       Yes
     LDAP_ProtocolVersion         3
     LDAP_SchemaVersion           3
     SearchPolicyBaseDN           o=ITSO,c=US
}

TcpImage TCPIPB NOFLUSH NOPURGE 1800
```

Figure 10-7 PAGENT configuration file

This file is the PAGENT configuration file. For more information on the PAGENT configuration file you can reference “The Policy Configuration File” in *z/OS V1R2 Communications Server: IP Configuration Reference*, SC31-8776. This information must be manually transferred (for example, FTPed, cut and pasted, or retyped) to the configuration file located on the zOS

V1R2.0 system. Typically the file is located in /etc/pagent.conf file and is used when the PAGENT application is started.

10.4.3 Work with IDS objects/rules

This section specifies the IDS policy rules. This is the most critical task and typically will be done iteratively until the final policy rules are defined.

- You are required to establish one Condition Set and one action set in at least one policy rule.
- You may optionally specify that rules apply only during validity periods.
- You may optionally associate rules with keywords to speed up their retrieval from LDAP.

Use this section to specify IDS policy rules, which can include Condition Sets, actions, policy keyword sets, or validity periods. Only one policy rule and associated actions can be applied to a particular packet.

The first step in creating a policy rule is to create the reusable objects that will be used in your action and Condition Sets. Next, create the actions and conditions based on the reuseable objects. Finally, build your policy rule from the available condition and action sets. The following sections walk you through this process.

When you have finished specifying IDS policy rules, select **File -> Send to LDAP** to store the policy information into the LDAP Server.

Even before you have finished specifying all the policies, you can save the (possibly incomplete) information that you have entered in an XML file on the workstation running zIDS Manager by selecting **File -> Save**. If you select **File -> Save As** you then have the option to save as an XML, LDIF, or CONF file.

Note: Only XML files can be read back in the zIDS Manager and edited. LDIF and CONF files are generated and are not reuseable by the zIDS Manager.

Work with reuseable objects

The reuseable objects are designed to be incorporated into multiple policies, conditions, or actions, depending on the type of object. Figure 10-8 illustrates the various object sets available in the Work with Reuseable Objects folder. The objects are stored in an object set and the collection of object sets is shown in the folder’s with the prefix All followed by the object set identifier. For example, all of the unique IP address sets created are located in the All IP Address Sets folder.

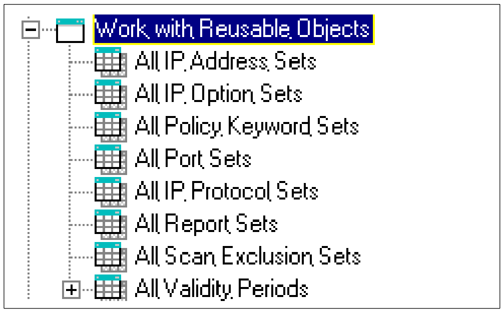


Figure 10-8 Reuseable objects

Let us create an IP address set. First, right-click the **All IP Address Sets** folder. You will see the screen shown in Figure 10-9.

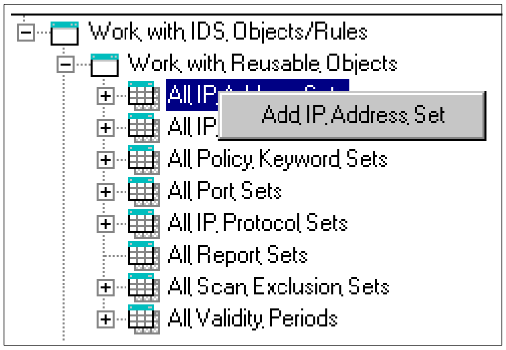
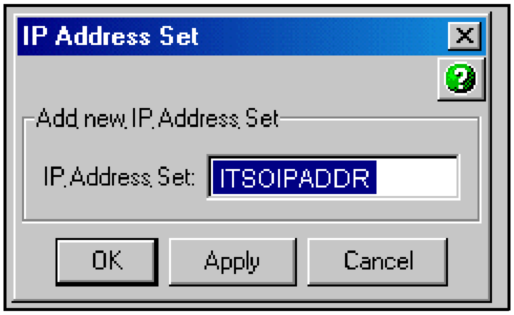


Figure 10-9 Add IP address set

Select **Add IP Address Set**. You will now be prompted for a name of the IP address set. In our example we called the object set ITSOIPADDR, as shown in Figure 10-10. Select **OK** and the new object set has been created. Notice that you can view the object sets either in the left pane by expanding the All IP Address Sets folder or the right pane through the summary information, as shown in Figure 10-11 on page 210.



IP Address Set: ITSOIPADDR

Figure 10-10 IP address set name

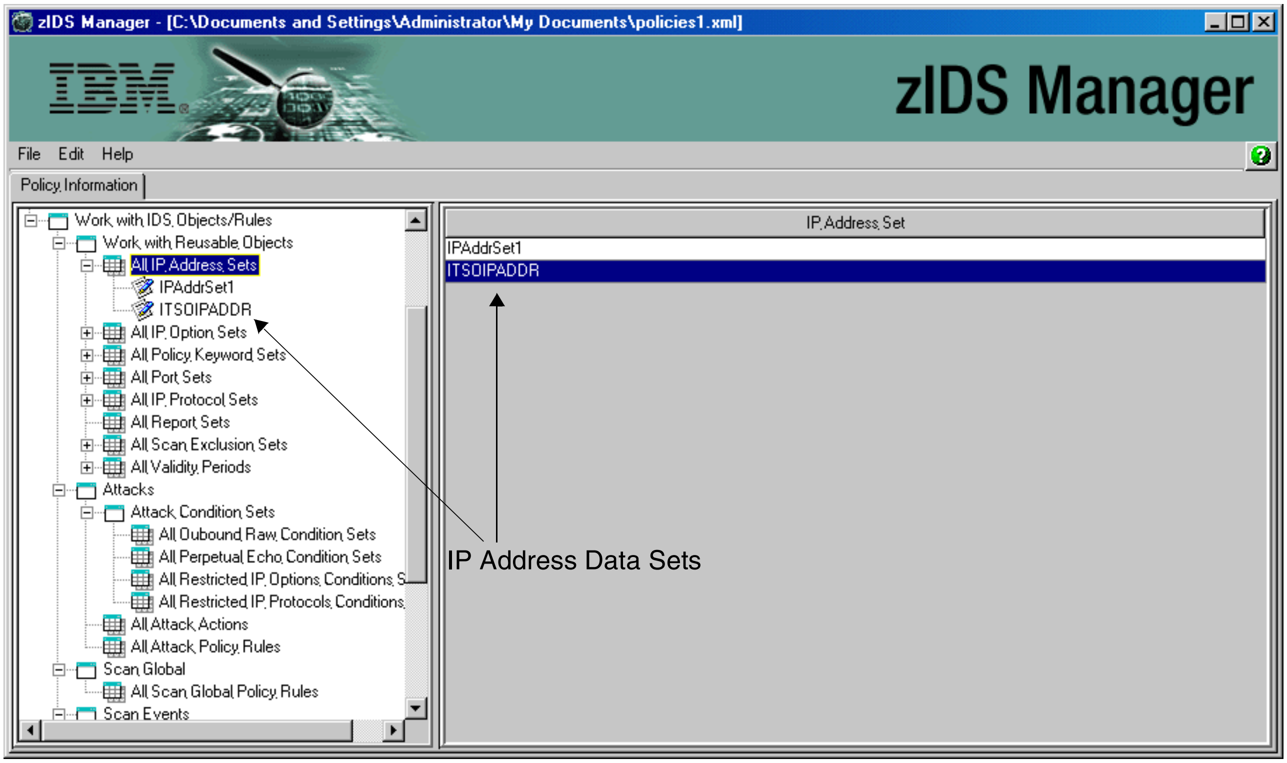
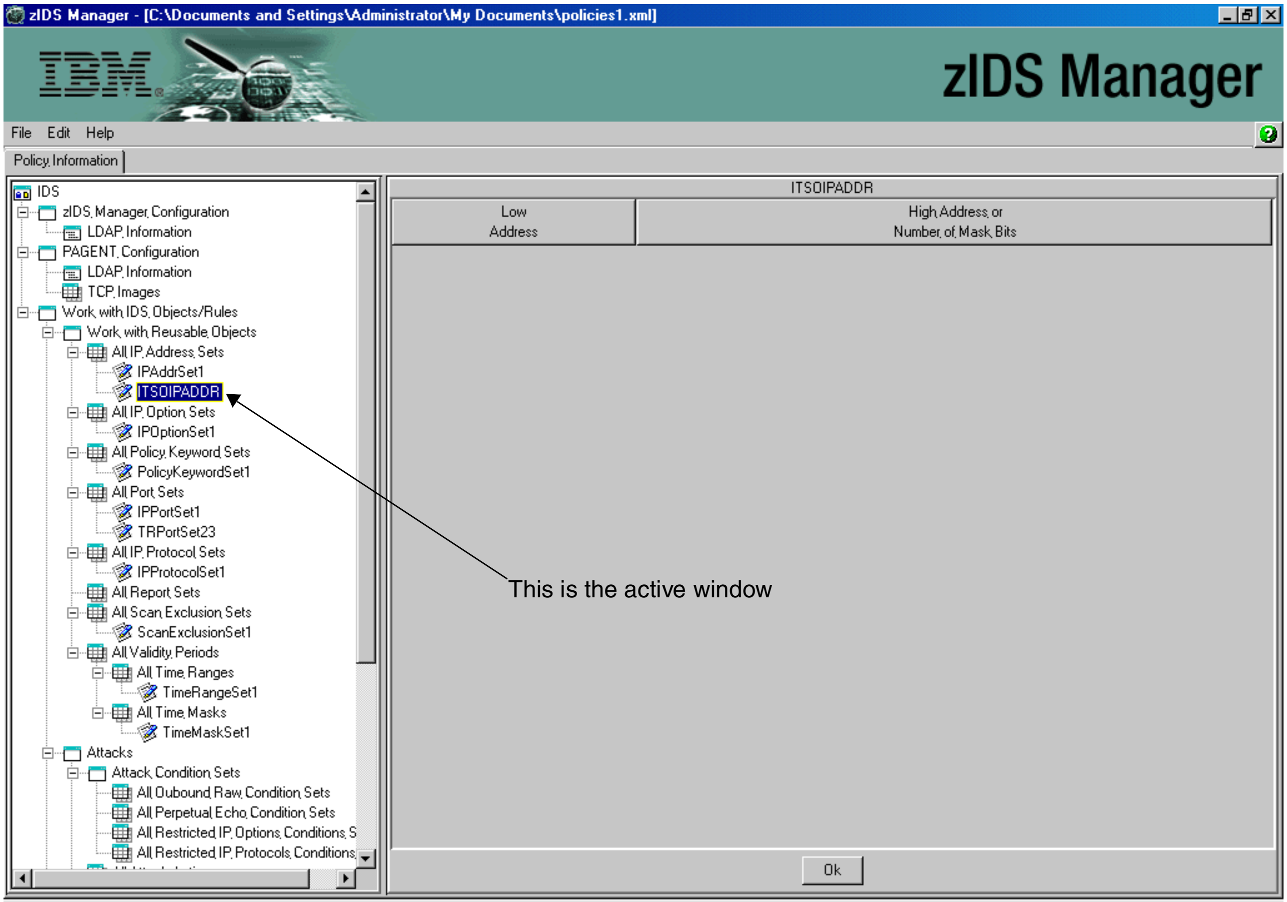


Figure 10-11 IP address sets

In this scenario there was already a predefined IP address set (that is, object set) named IPAddrSet1. Next we want to modify the information that is in the object set ITSOIPADDR.

Click the ITSOIPADDR object set making this the active element in the left pane. The item that is highlighted in the left pane is the active window and dictates the information you see in the right pane, as shown in Figure 10-12 on page 211.



This is the active window

Figure 10-12 IPAddr object set ITSOIPADDR

Now right-click and the pop-up menu appears, as shown in Figure 10-13 on page 212.

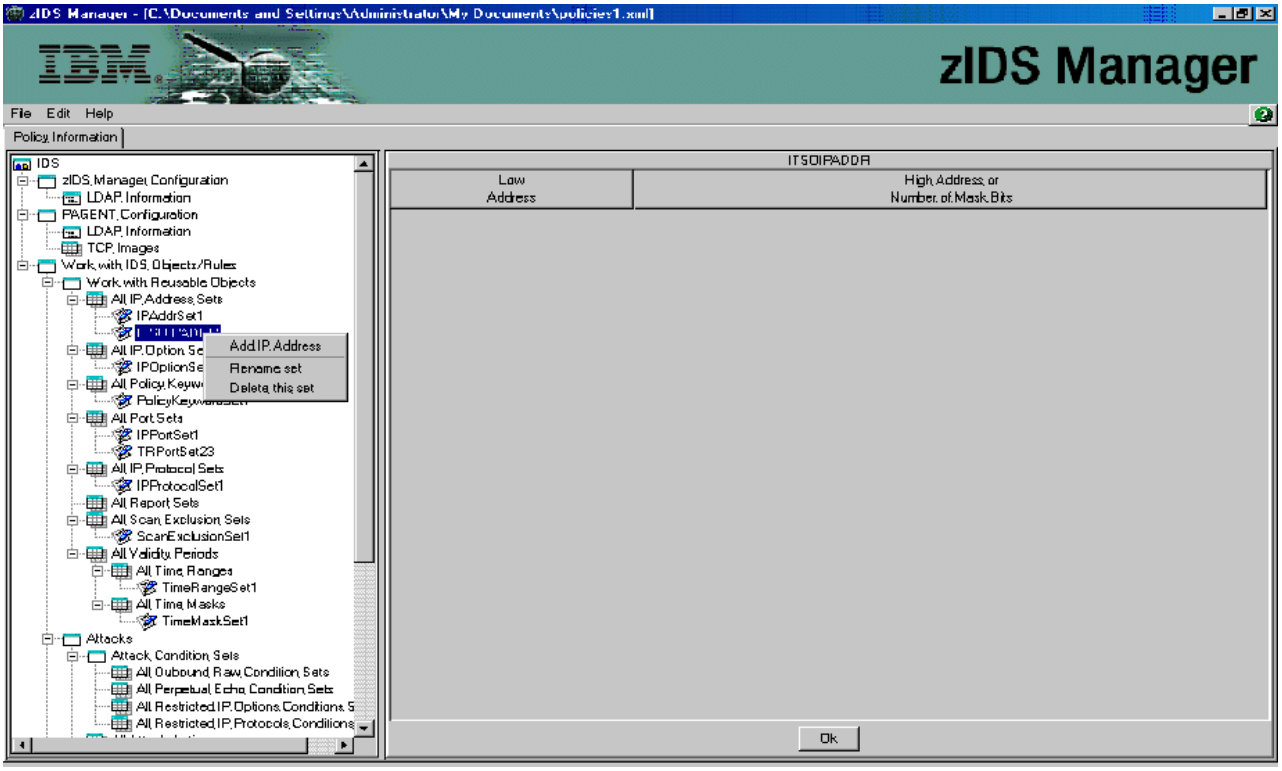


Figure 10-13 Object set options

The available options are:

Add IP address Select this option to enter a low IP address and a high IP address or number of mask bits.

Delete this set Select this option to delete the highlighted IP address set and all IP address ranges in the set. The deleted IP address set will be removed from all associated Condition Sets.

Rename set Select this option to rename the highlighted IP address set. The original IP address set name will be removed from all associated Condition Sets. The new set name will have to be re-associated with the desired Condition Sets.

Select **Add IP Address** and the box shown in Figure 10-14 appears.

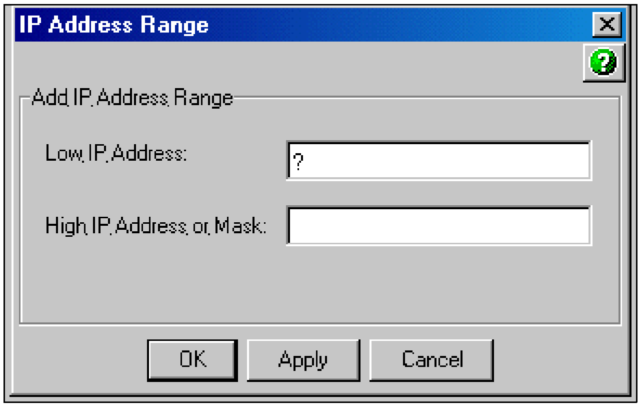
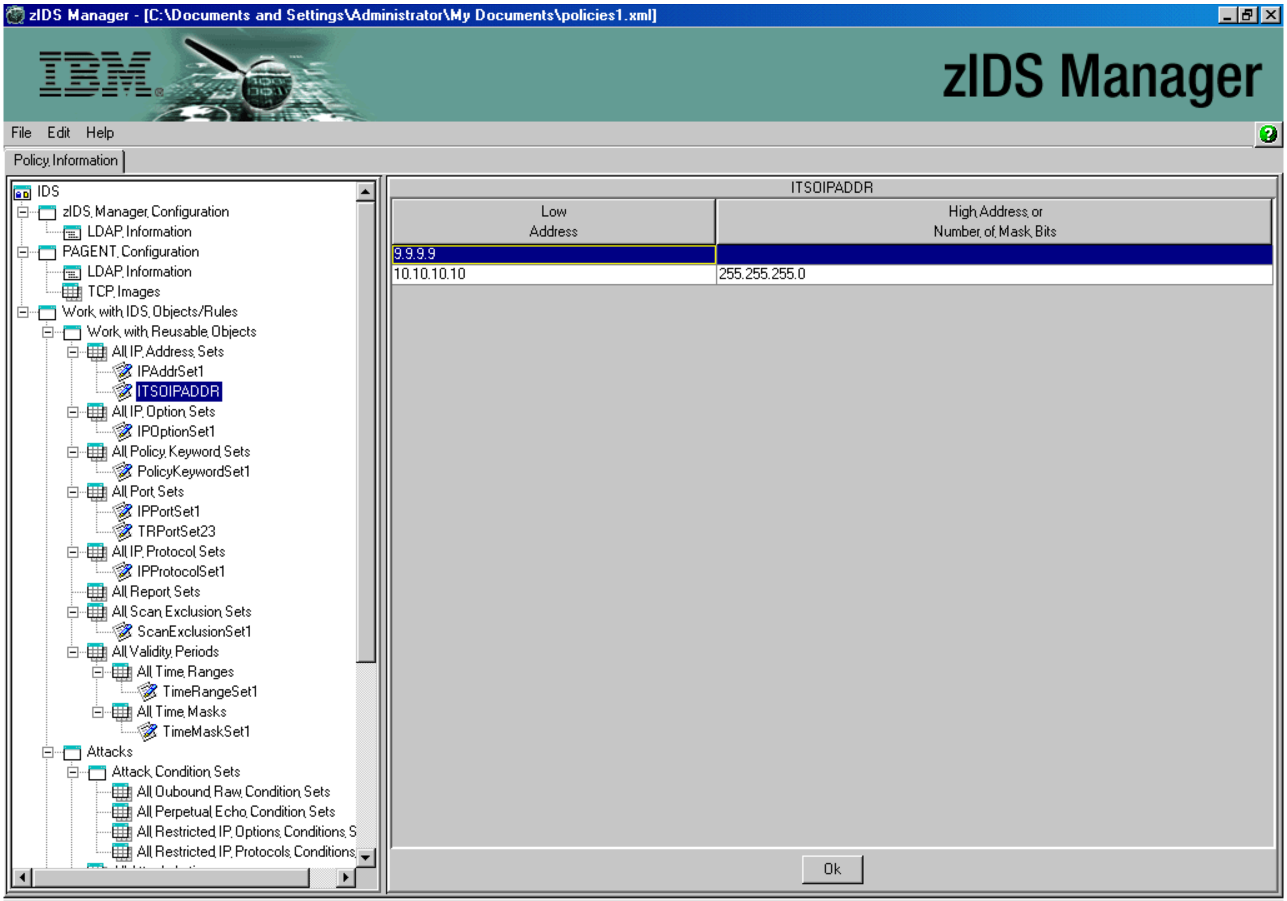


Figure 10-14 IP address range

The question mark indicates that this field requires a value. In our case, we place the value 9.9.9.9 in the Low IP Address field and click **Apply**, followed by another low IP address of 10.10.10.10 and a high IP address or number of mask bits of 24. Select **OK**. We have successfully created a reusable object. The ITSOIPADDR object set contains two objects, as shown in Figure 10-15.

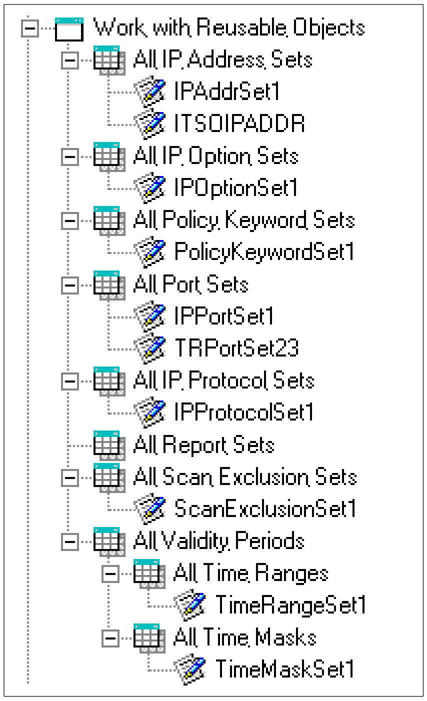


ITSOIPADDR

Low Address	High Address or Number of Mask Bits
9.9.9.9	
10.10.10.10	255.255.255.0

Figure 10-15 Object set summary information

The process by which we built this object set is the same for all of the reuseable objects. Of course, there are different parameters depending on the type of object set being constructed (that is, port sets require a port). We have built several other reusable objects that will be used at a later time as per Figure 10-16 on page 214. It is recommended at this time that the reader create reusable object sets with the same names (that follow) for condition, action, and policy building.



- Work with Reusable Objects
  - All IP Address Sets
    - IPAddrSet1
    - ITSOIPADDR
  - All IP Option Sets
    - IPOptionSet1
  - All Policy Keyword Sets
    - PolicyKeywordSet1
  - All Port Sets
    - IPPortSet1
    - TRPortSet23
  - All IP Protocol Sets
    - IPProtocolSet1
  - All Report Sets
  - All Scan Exclusion Sets
    - ScanExclusionSet1
  - All Validity Periods
    - All Time Ranges
      - TimeRangeSet1
    - All Time Masks
      - TimeMaskSet1

Figure 10-16 Constructed reusable objects

Attacks

An attack can be a single packet designed to crash or hang a system, or multiple packets designed to consume a limited resource causing a network, system, or application to be unavailable to its intended users (a denial of service). IDS attack policy lets you turn on attack detection for one or more categories of attacks independently of each other. In general, the types of actions that you can specify for an attack policy are event logging, statistics gathering, packet tracing, and discarding of attack packets.

The next step is for us to generate an attack condition, action, and policy. Let us start with the condition. Click **Attack Condition Set**, making this the active folder. The folder is open if the + or - next to the folder is a -. You should see the same screen shown in Figure 10-17 on page 215.

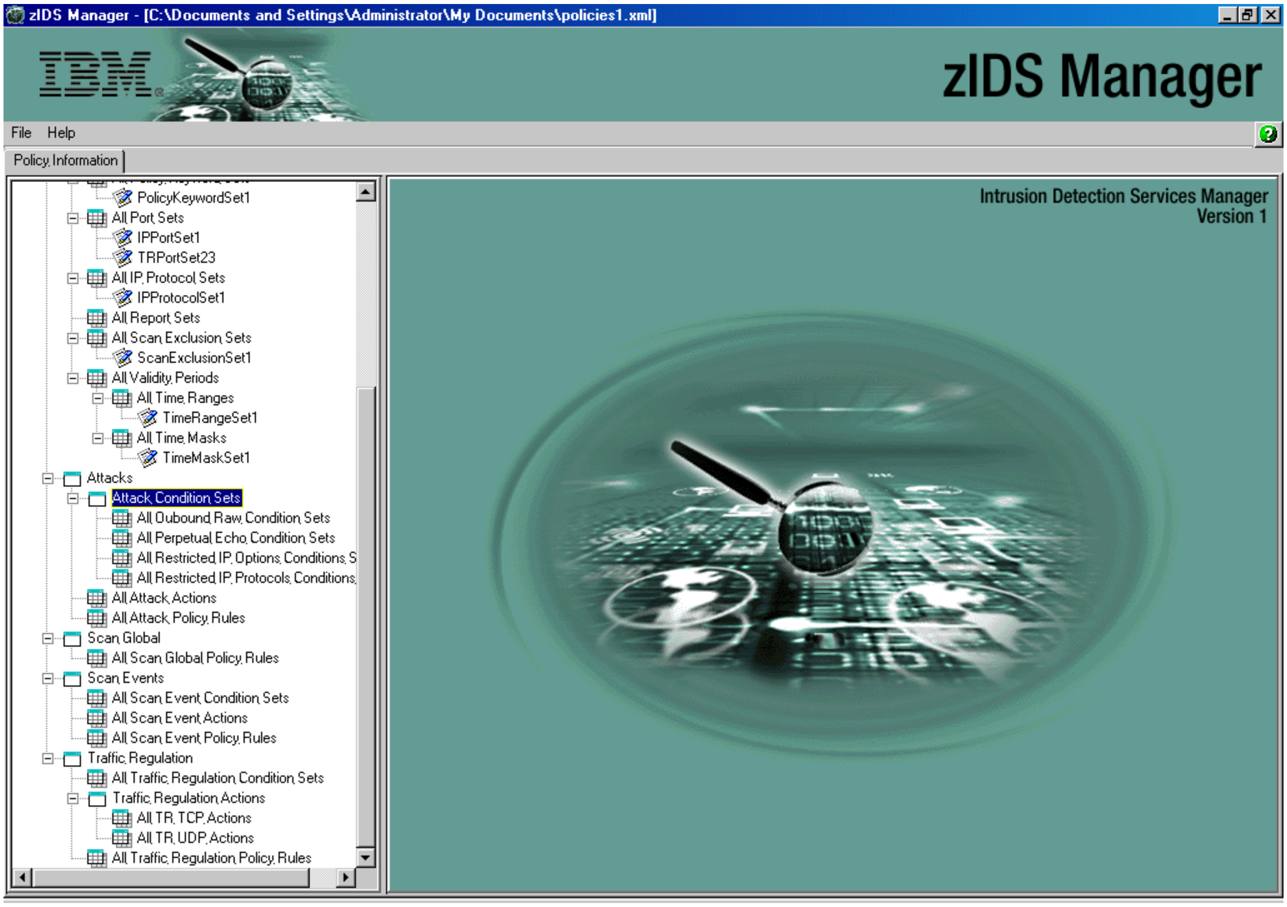
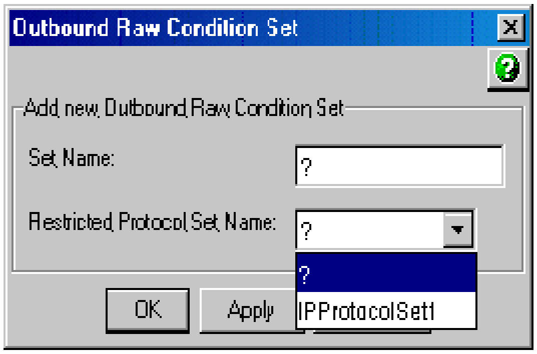

zIDS Manager - [C:\Documents and Settings\Administrator\My Documents\policies1.xml]
IBM
zIDS Manager
File
Help
Policy Information
PolicyKeywordSet1
All Port Sets
IPPortSet1
TRPortSet23
All IP Protocol Sets
IPProtocolSet1
All Report Sets
All Scan Exclusion Sets
ScanExclusionSet1
All Validity Periods
All Time Ranges
TimeRangeSet1
All Time Masks
TimeMaskSet1
Attacks
Attack Condition Sets
All Oubound Raw Condition Sets
All Perpetual Echo Condition Sets
All Restricted IP Options Conditions S
All Restricted IP Protocols Conditions
All Attack Actions
All Attack Policy Rules
Scan Global
All Scan Global Policy Rules
Scan Events
All Scan Event Condition Sets
All Scan Event Actions
All Scan Event Policy Rules
Traffic Regulation
All Traffic Regulation Condition Sets
Traffic Regulation Actions
All TR TCP Actions
All TR UDP Actions
All Traffic Regulation Policy Rules
Intrusion Detection Services Manager
Version 1

Figure 10-17 Attack Condition Sets

Only four attack types (outbound raw, perpetual echo, restricted IP options, and restricted IP protocols) have Condition Sets for which the user can specify values. Select **All Outbound Raw Condition Sets**, making it active in the left pane, followed by right-clicking to bring up the option to **Add Outbound Raw Condtion Set**. Select this option by clicking it and the menu appears as shown in Figure 10-18.

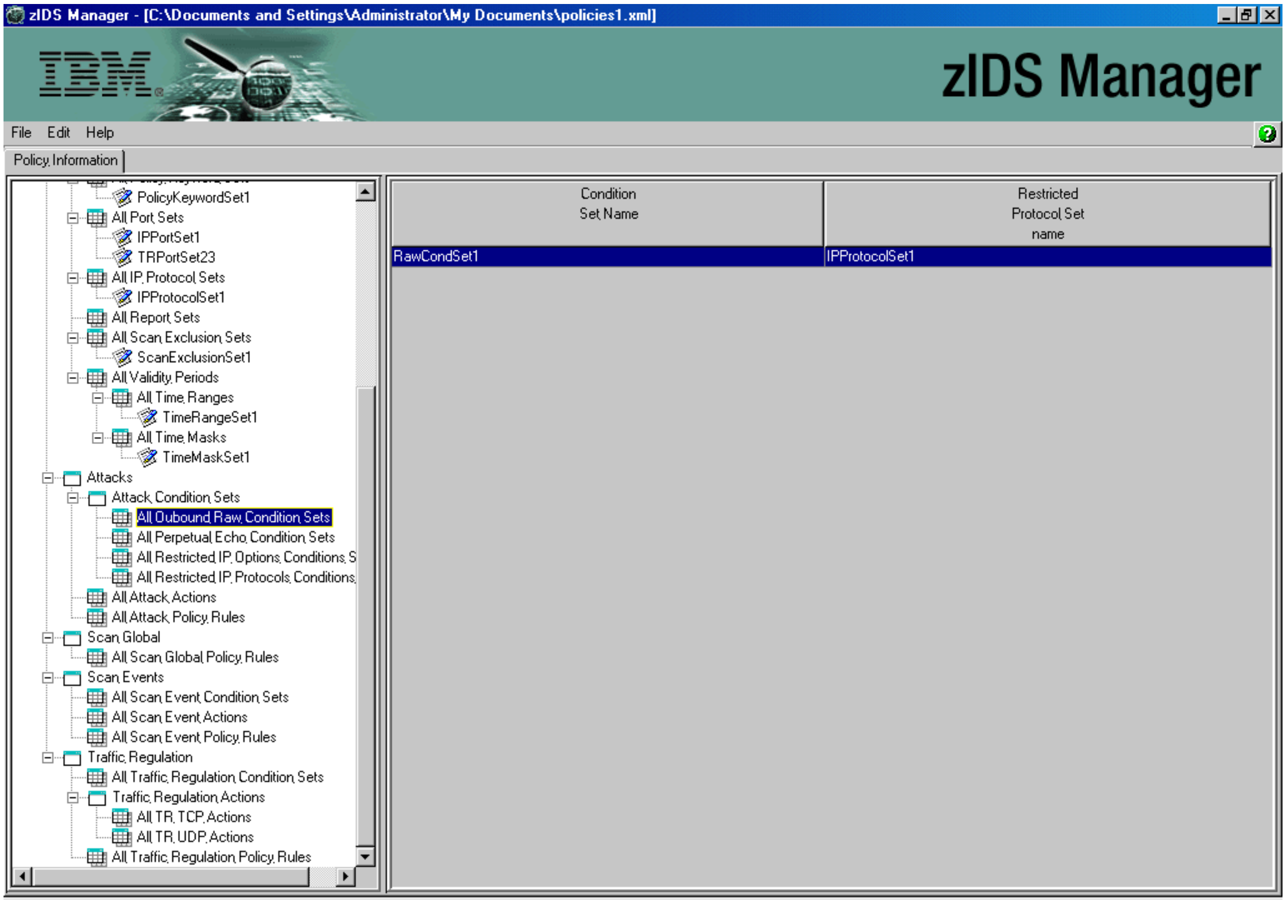


Set Name: ?

Restricted Protocol Set Name: ?

Figure 10-18 Outbound Raw Condition Set

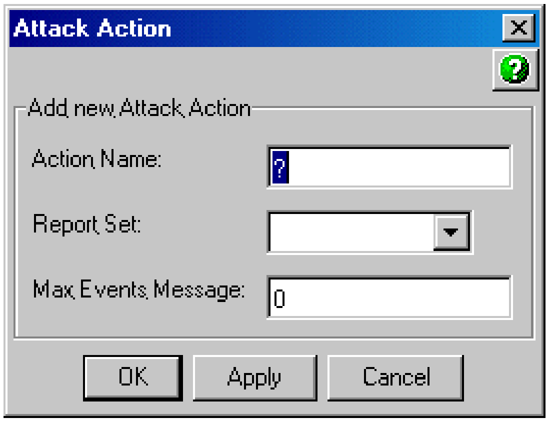
The set name is the name associated with this condition. We chose to use the name RawCondSet1. The **Restricted Protocol Set Name** option presents us with a drop-down menu that includes all of the objects built in the **Reusable Object/All IP Protocol Sets**. Let us select the **IPProtocolSet1** object. Select **OK**. We have generated a Condition Set, as shown in Figure 10-19.



Condition Set Name	Restricted Protocol Set name
RawCondSet1	IPProtocolSet1

Figure 10-19 Summary of raw Condition Sets

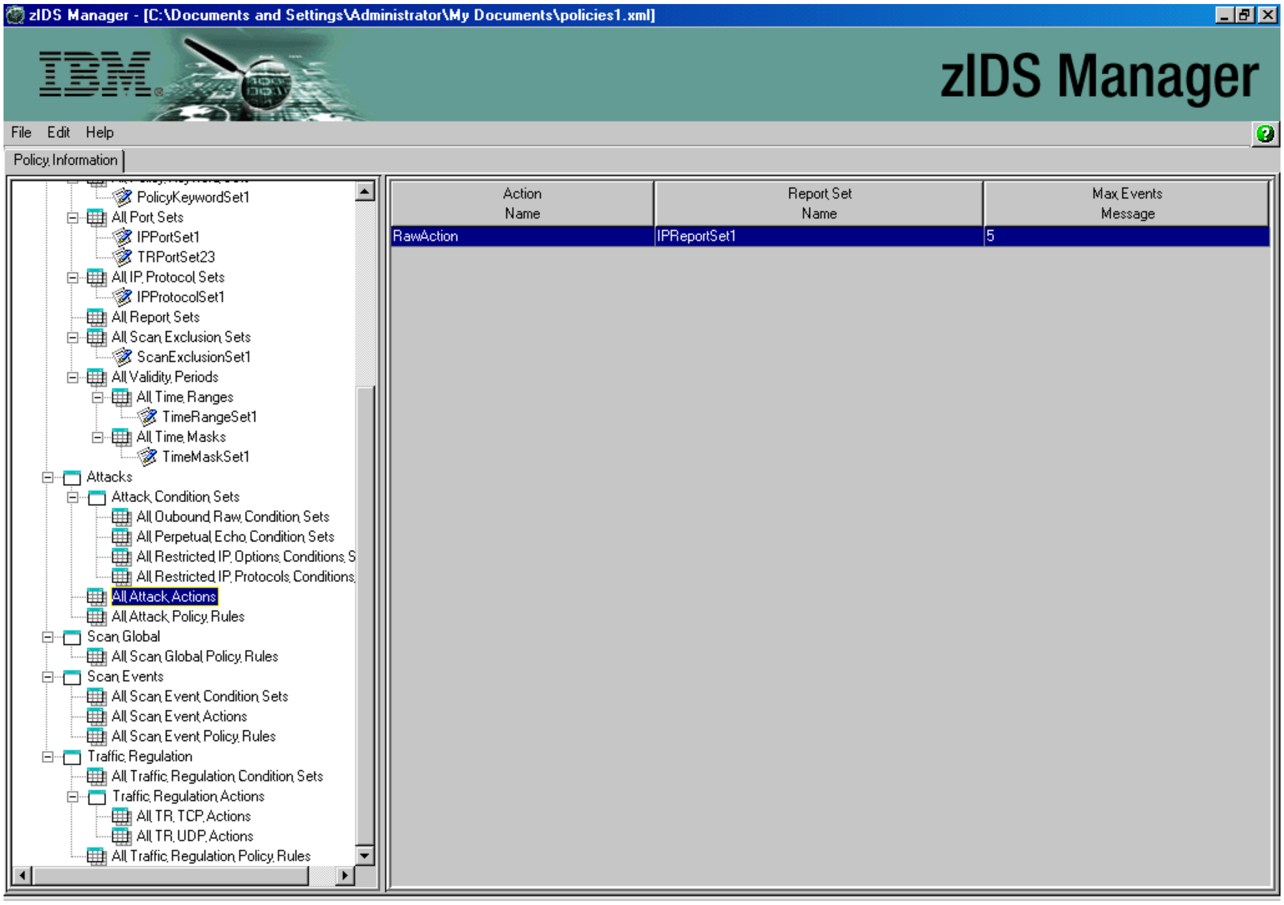
Next select **All Attack Actions** in the left pane; then right-click and select **Add Attack Actions**. The pop-up menu appears as shown in Figure 10-20.



Action Name:	?
Report Set:	
Max Events Message:	0

Figure 10-20 Attack action pop-up menu

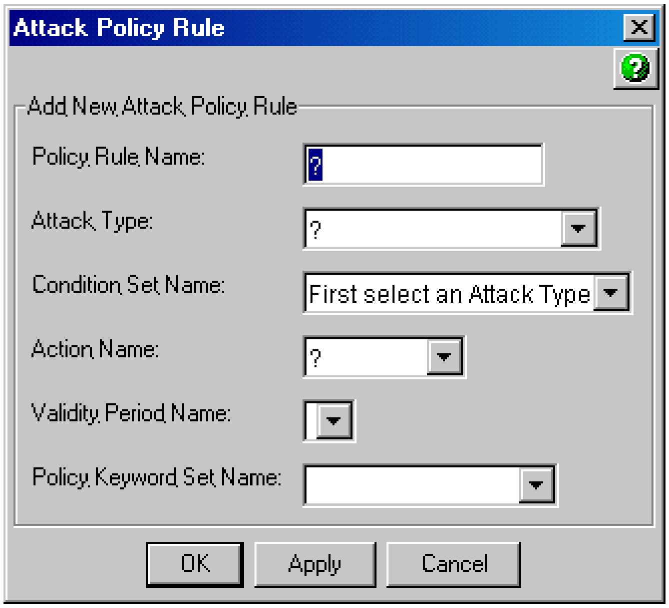
Action Name is a required field and we chose the name RawAction. **Report Set** is an optional field with a drop-down menu containing the reusable report sets. In our case, we will use IPReportSet1. Select **OK**. There is now attack action built, as shown in Figure 10-21.



Action Name	Report Set Name	Max Events Message
RawAction	IPReportSet1	5

Figure 10-21 Attack action summary

There now needs to be a policy associating a condtion with an action. Select **All Attack Policy Rules** by clicking the folder. Now, right-click and select **Add Attack Policy Rule/Below Section**. The pop-up menu appears as shown in Figure 10-22 on page 218.



Attack Policy Rule

Add New Attack Policy Rule

Policy Rule Name: ?

Attack Type: ?

Condition Set Name: First select an Attack Type

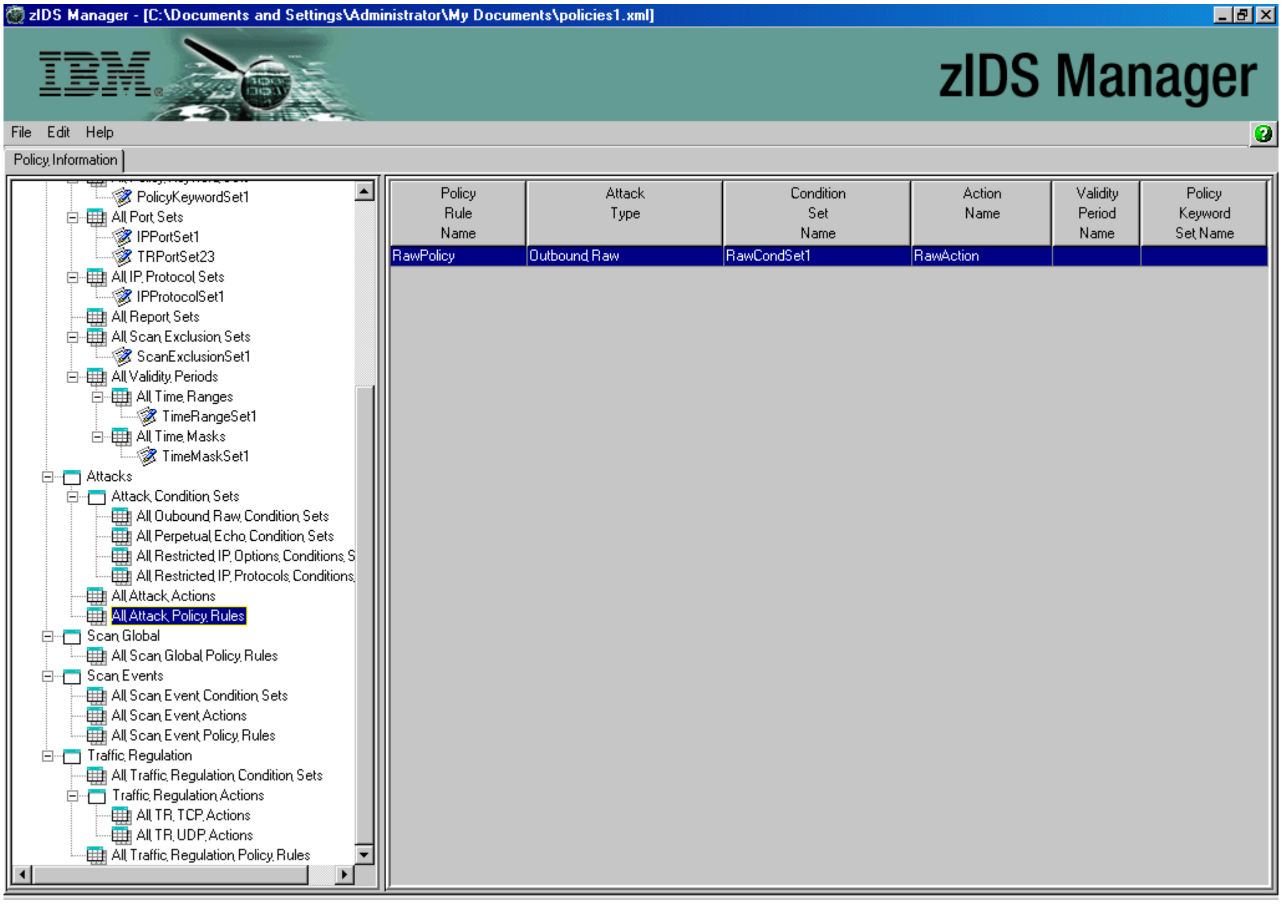
Action Name: ?

Validity Period Name:

Policy Keyword Set Name:

Figure 10-22 Attack policy rule pop-up menu

Policy Rule Name, **Attack Type**, **Condition Set Name**, and **Action Name** are all required fields. We will use RawPolicy, Outbound Raw, RawCondSet1, and RawAction, respectively. **Attack Type** has a drop-down menu. This contains the different attack categories; see 9.2, “Attack policies” on page 193, for information on the attack types or place the cursor in this field and press the F1 key. The **Validity Period Name** and **Policy Keyword Set Name** fields are optional. Other fields and are not used in this example. Select **OK**. An attack policy has been built for outbound raw sockets, as shown in Figure 10-23 on page 219.



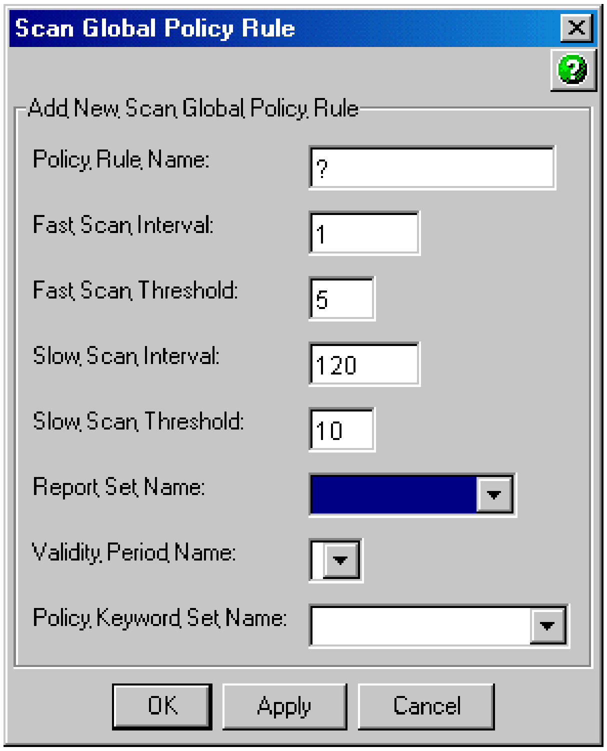
Policy Rule Name	Attack Type	Condition Set Name	Action Name	Validity Period Name	Policy Keyword Set Name
RawPolicy	Outbound Raw	RawCondSet1	RawAction		

Figure 10-23 Attack policy

Scan global

You can specify sets of global scan detection parameters (threshold and interval for fast and slow scans). These attributes apply to all scan events. If you configure a certain category of scan event, the action will be triggered if the number of those events received from one IP address exceeds the slow scan threshold during the slow scan interval. Similarly, if you configure a certain category of scan event, the action will be triggered if the number of those events received from one IP address exceeds the fast scan threshold during the fast scan interval. The slow scan threshold must be greater than the fast scan threshold. The slow scan interval must be greater than the fast scan interval.

Open the **Scan Global** folder in the left pane. Click **All Scan Global Policy Rules**. Next, right-click and select **Add Scan Global Policy Rule/Below Section**. The menu appears as shown in Figure 10-24 on page 220.



Scan Global Policy Rule

Add New Scan Global Policy Rule

Policy Rule Name: ?

Fast Scan Interval: 1

Fast Scan Threshold: 5

Slow Scan Interval: 120

Slow Scan Threshold: 10

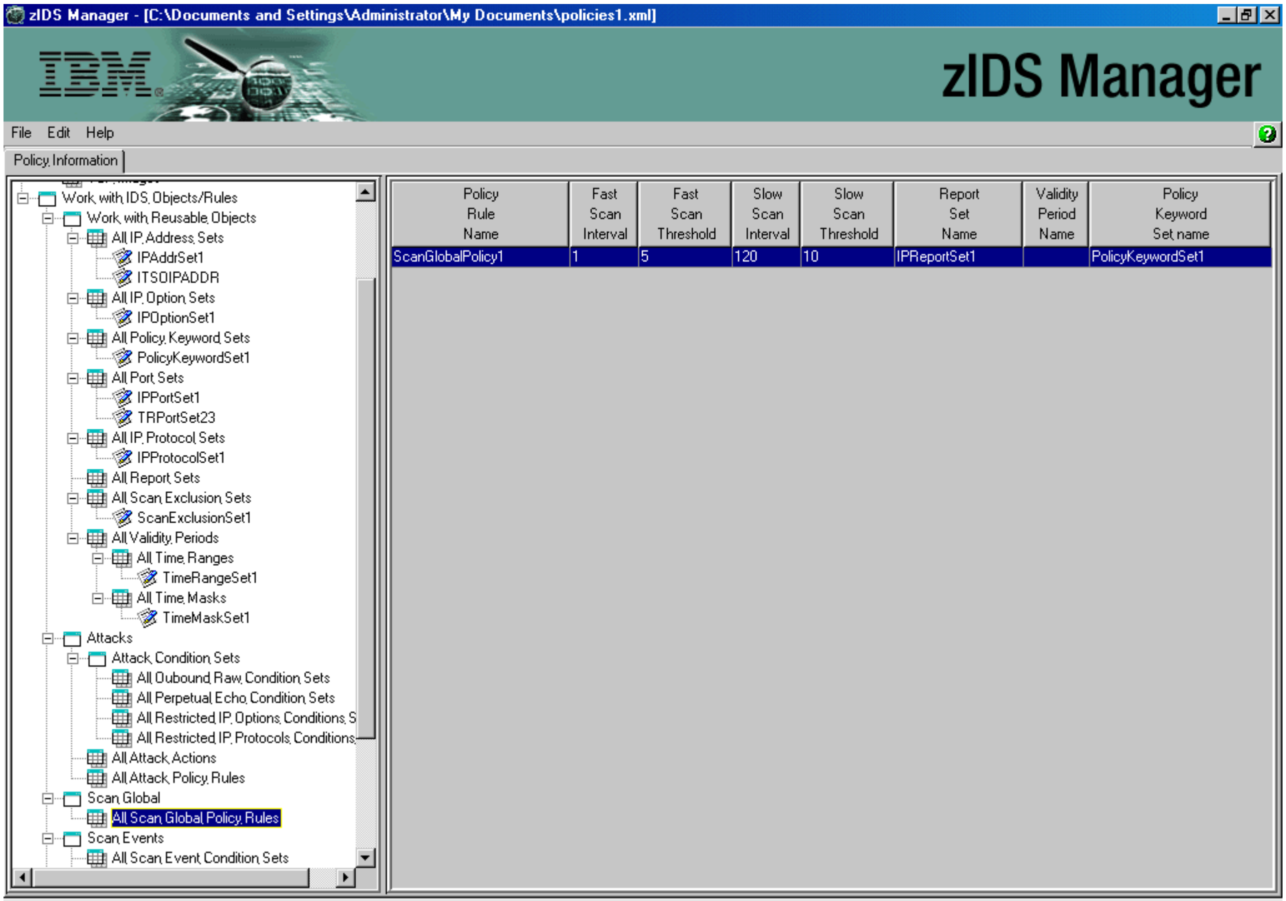
Report Set Name:

Validity Period Name:

Policy Keyword Set Name:

Figure 10-24   Scan global policy rule pop-up menu

As we have already seen, there needs to be a policy name. For the policy rule name we used the name ScanGlobalPolicy1. Notice that there are default values for **Fast Scan Interval**, **Fast Scan Threshold**, **Slow Scan Interval**, and **Slow Scan Threshold** fields. We will accept all of the default values. The **Report Set Name** and **Policy Keyword Set Name** fields have drop-down menus indicating that the choices are defined as reusable objects. We chose IPReportSet1 and PolicyKeywordSet1, respectively. Select **OK**. A scan global policy is now created, as shown in Figure 10-25 on page 221.



Policy Rule Name	Fast Scan Interval	Fast Scan Threshold	Slow Scan Interval	Slow Scan Threshold	Report Set Name	Validity Period Name	Policy Keyword Set name
ScanGlobalPolicy1	1	5	120	10	IPReportSet1		PolicyKeywordSet1

Figure 10-25 Scan global policy

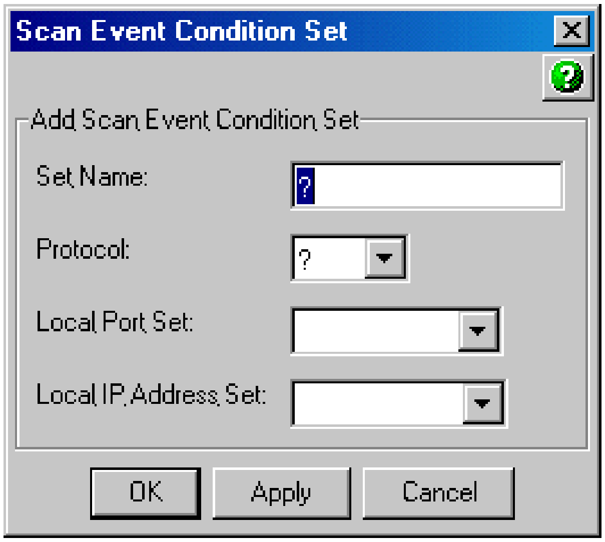
Scan events

Scan events come from the following categories:

- ICMP scans: ICMP requests (echo, information, timestamp, and subnet mask) are used to map network topology. Any request sent to a subnet base or broadcast address will be treated as very suspicious. Echo requests (PING) and timestamp requests are normal unless they include the Record Route or Record Timestamp option, in which case they are possibly suspicious.
- TCP port scans: Because TCP is a stateful protocol, many different events may be classified as normal, suspicious, or highly suspicious. For more details, please see the section "Scan policies" of *OS/390 V2R10.0 IBM Communications Server IP Configuration Guide,* SC-31-8725.
- UDP port scans: A datagram received for a restricted port is very suspicious, one received for an unreserved but unbound port is possibly suspicious, and one received for a bound port is normal.

The individual packets used in a scan can be categorized as normal, possibly suspicious, or very suspicious. To control the performance impact and analysis load of scan monitoring, you can adjust your interest level in potential scan events. If you set the sensitivity level to high, then normal, possibly suspicious, and very suspicious events will be counted. If you set the sensitivity level to medium, then possibly suspicious and very suspicious events will be counted. If you set the sensitivity level to low, then only very suspicious events will be counted. If you set the sensitivity level to none, then no events will be counted.

First open the **Scan Events** folder. Click **All Scan Event Conditions Sets** to activate. Right-click and select **Add Scan Event Condtion Set** (see Figure 10-26).



Add Scan Event Condition Set

Set Name: ?

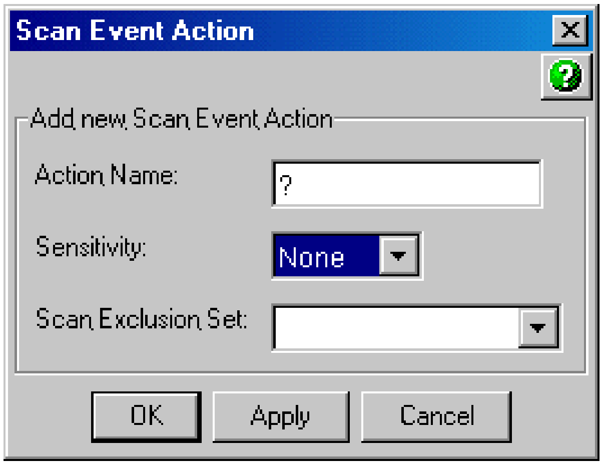
Protocol: ?

Local Port Set:

Local IP Address Set:

Figure 10-26 Scan event Condition Set pop-up menu

The required fields are **Set Name** and **Protocol**. We will use ScanEventCondition1 and TCP, respectively. The **Local Port Set** and **Local IP Address Set** fields have drop-down menus that contain reusable objects. For these fields we will use the reusable objects IPPortSet1 and IPAddrSet1. Next select **OK** and we have created a condtion set. Now we need an action and, as with the Attack section, a policy to relate the condtion to an action. Click **All Scan Event Actions** in the left pane, making it the active element. Right-click and select **Add Scan Event Action**. The pop-up menu appears, as shown in Figure 10-27.



Add new Scan Event Action

Action Name: ?

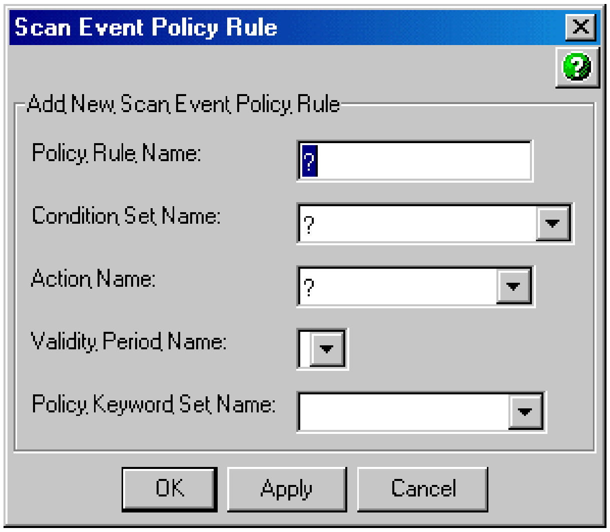
Sensitivity: None

Scan Exclusion Set:

Figure 10-27 Scan event action pop-up menu

Action Name is a required field and our action is called ScanEventAction1. The sensitivity defaults to None. This will cause no events to be counted toward the scan event. We chose a low sensitivity. For more information on the sensitivity, place your cursor in the **Sensitivity** field and press the F1 key. **Scan Exclusion Set** is optional, and for this action we will not code a value. Select **OK** to complete the action.

Next click **All Scan Event Policy Rules**, making this the active folder. Right-click and choose **Add Scan Event Policy Rule -> Below Section**. The pop-up menu appears as shown in Figure 10-28.



Policy Rule Name: ?

Condition Set Name: ?

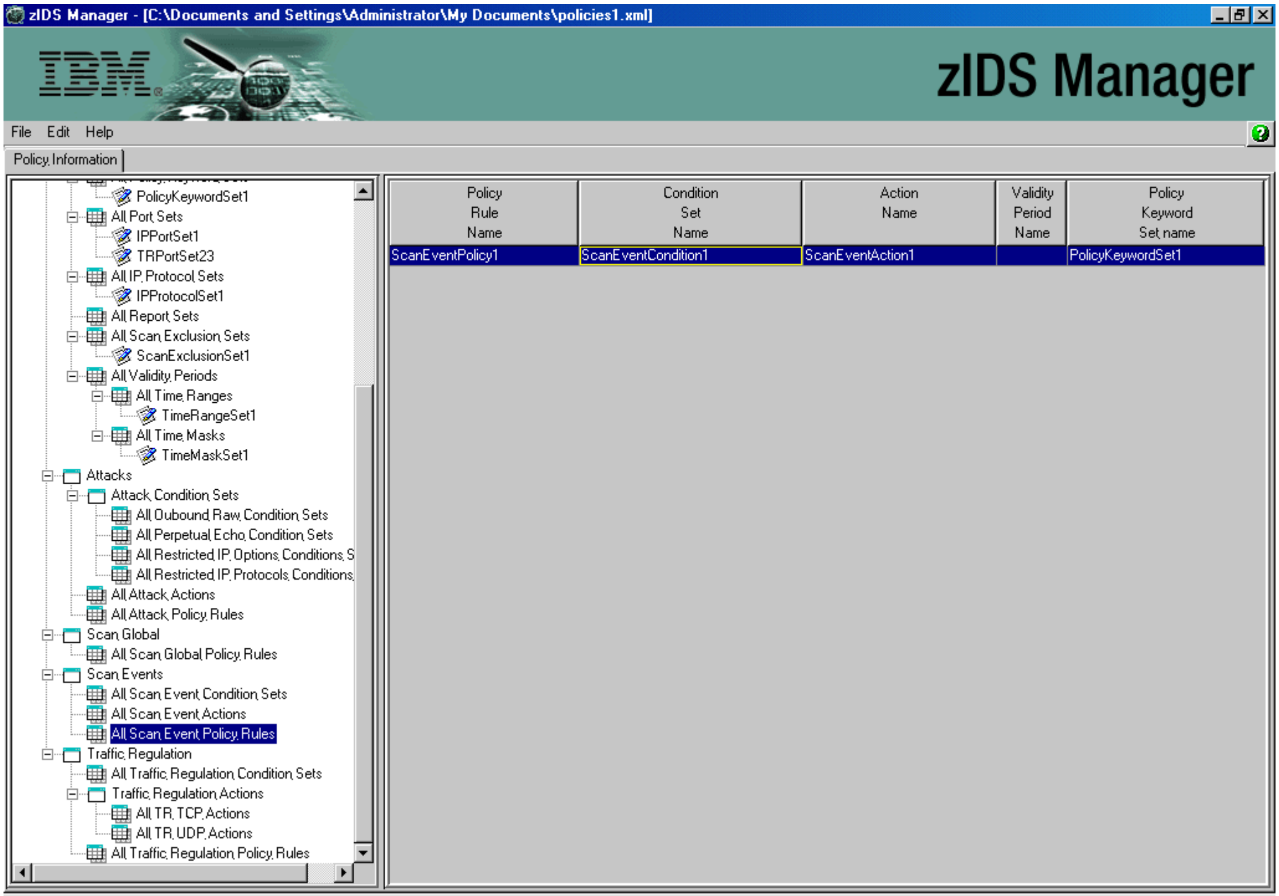
Action Name: ?

Validity Period Name:

Policy Keyword Set Name:

Figure 10-28 Scan event policy rule pop-up menu

The required fields are **Policy Rule Name**, **Condition Set Name**, and **Action Name**. We will use ScanEventPolicy1, ScanEventCondition1, and ScanEventAction1, respectively. **Validty Period Name** and **Policy Keyword Name** are optional fields that will be left blank. Select **OK** and our policy is now complete, as shown in Figure 10-29 on page 224. To modify the policy, you can double-click the policy in the right pane.



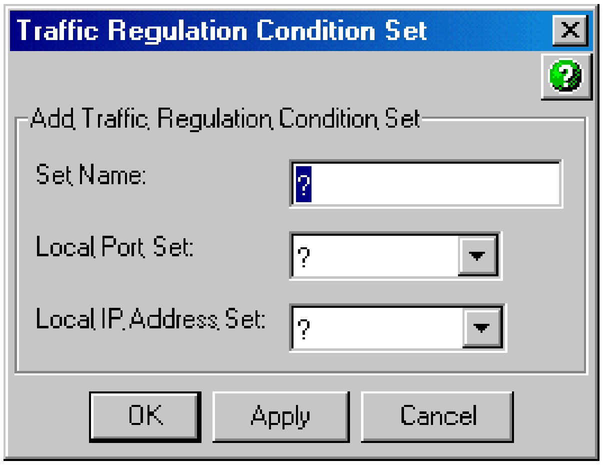
Policy Rule Name	Condition Set Name	Action Name	Validity Period Name	Policy Keyword Set name
ScanEventPolicy1	ScanEventCondition1	ScanEventAction1		PolicyKeywordSet1

Figure 10-29 Scan event policy rule

Traffic Regulation

Traffic Regulation (TR) policies are used to limit memory resource consumption and queue delay during peak loads. TR policies for TCP ports can limit the total number of connections an application has active at one time. This can be used to limit the number of address spaces created by forking applications such as FTPD and otelnetd. A fair share algorithm is also provided based on the percentage of remaining available connections held by a source IP address. IDS policies for UDP ports specify a queue length. Longer queues let applications with higher processing rate capacity absorb higher bursts of traffic. Shorter queues let applications with lower processing rate capacity reduce the queue delay time of packets that they accept.

First open the **Traffic Regulation** folder. Click **All Traffic Regulation Condition Sets** to activate it. Right-click and select **Add Traffic Regulation Condition Set** (see Figure 10-30 on page 225).



Traffic Regulation Condition Set

Add Traffic Regulation Condition Set

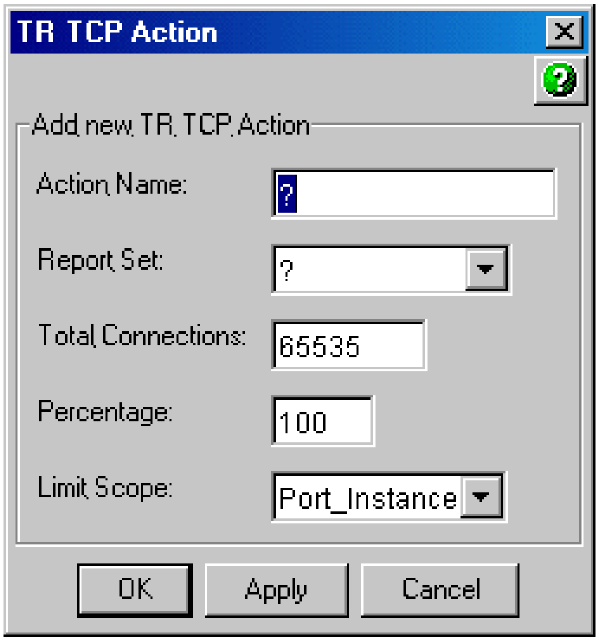
Set Name: ?

Local Port Set: ?

Local IP Address Set: ?

Figure 10-30 Traffic Regulation Condition Set pop-up menu

All of the fields are required as per the question marks. The set name is TRREGCondSet1. **Local Port Set** and **Local IP Address Set** have drop-down menus and we will use the TRPPortSet23 and IPAddrSet1 reuseable objects, respectively. Select **OK** and the conditions set is complete. Now click **All TR TCP Actions** (we will generate a TCP action) in the left pane, making it the active element. Right-click and select **Add TR TCP Action**. The pop-up menu appears as shown in Figure 10-31.



TR TCP Action

Add new TR TCP Action

Action Name: ?

Report Set: ?

Total Connections: 65535

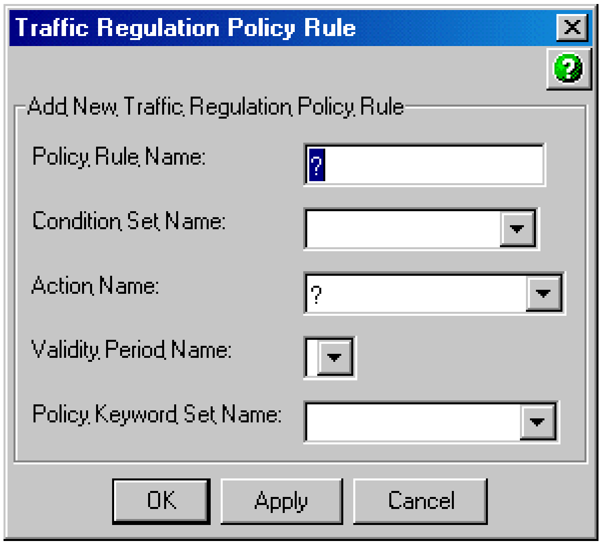
Percentage: 100

Limit Scope: Port_Instance

Figure 10-31 TR TCP action pop-up menu

Action Name is a required field and our action is called TRTCPAction1. **Report Set** is also required and we will use our reusable object report set TRReportSet1. We accept the default values of 65535, 100, and Port_Instance for **Total Connections, Percentage**, and **Limit Scope**, respectively. Finally, we must correlate the condtion with the action through a policy.

Next click **All Traffic Regulation Policy Rules** followed by a right-click. Now choose **Add Traffic Regulation Policy Rule -> Below Selection**. The pop-up menu appears as shown in Figure 10-32 on page 226.



Traffic Regulation Policy Rule

Add New Traffic Regulation Policy Rule

Policy Rule Name: ?

Condition Set Name:

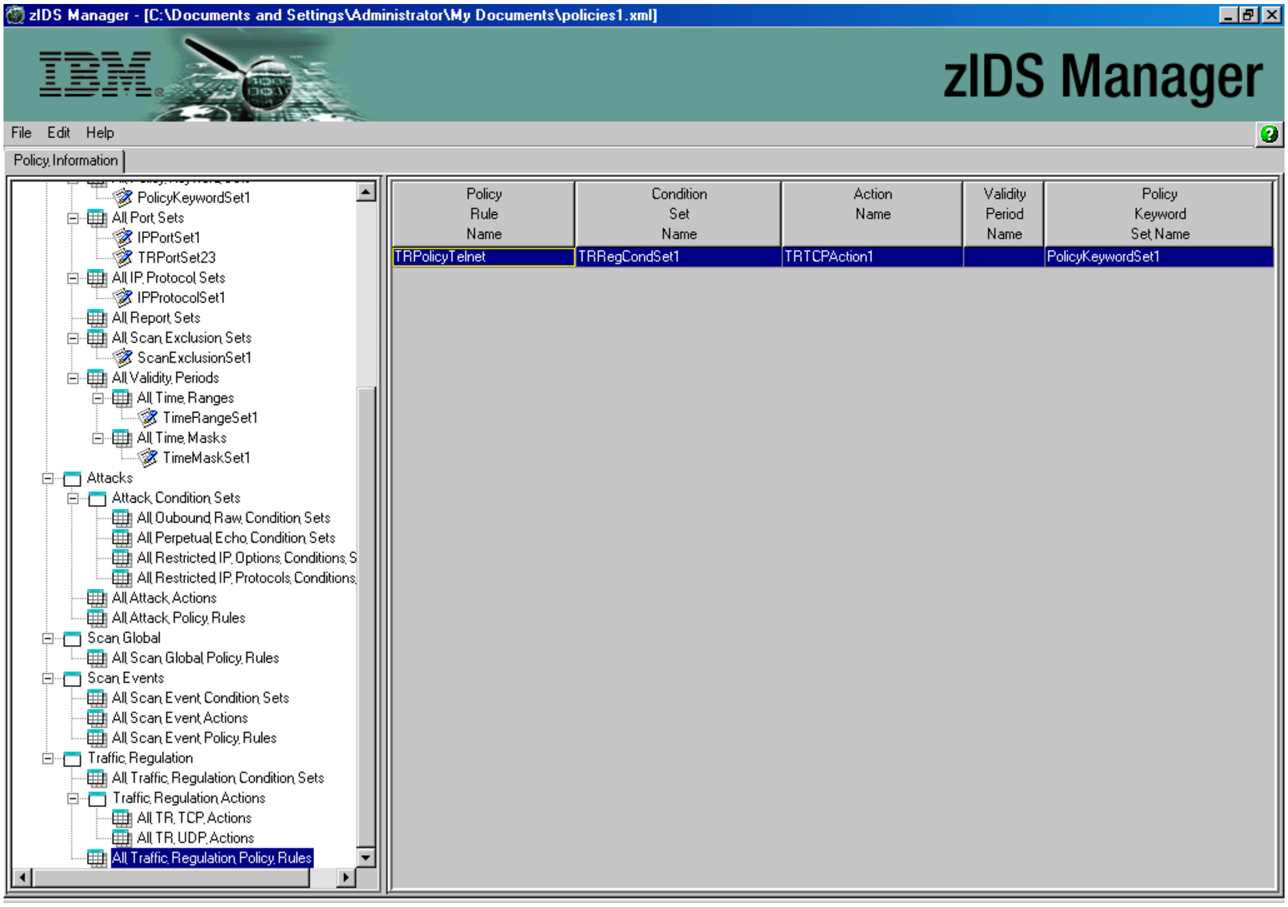
Action Name: ?

Validity Period Name:

Policy Keyword Set Name:

Figure 10-32 Traffic Regulation policy rule pop-up menu

The required fields are **Policy Rule Name**, **Conditions Set Name**, and **Action Name**. For **Policy Rule Name** we will use TRPolicyTelnet. **Condition Set Name** and **Action Name** are drop-down menus and we will use our reuseable objects, TRRegCondSet1 and TRTCPAction1, respectively. **Validty Period Name** and **Policy Keyword Name** are optional fields that will be left blank. Select **OK** and our policy is now complete, as shown in Figure 10-33 on page 227. To modify the policy, you can double-click the policy in the right pane.



Policy Rule Name	Condition Set Name	Action Name	Validity Period Name	Policy Keyword Set Name
TRPolicyTelnet	TRRegCondSet1	TRTCPAction1		PolicyKeywordSet1

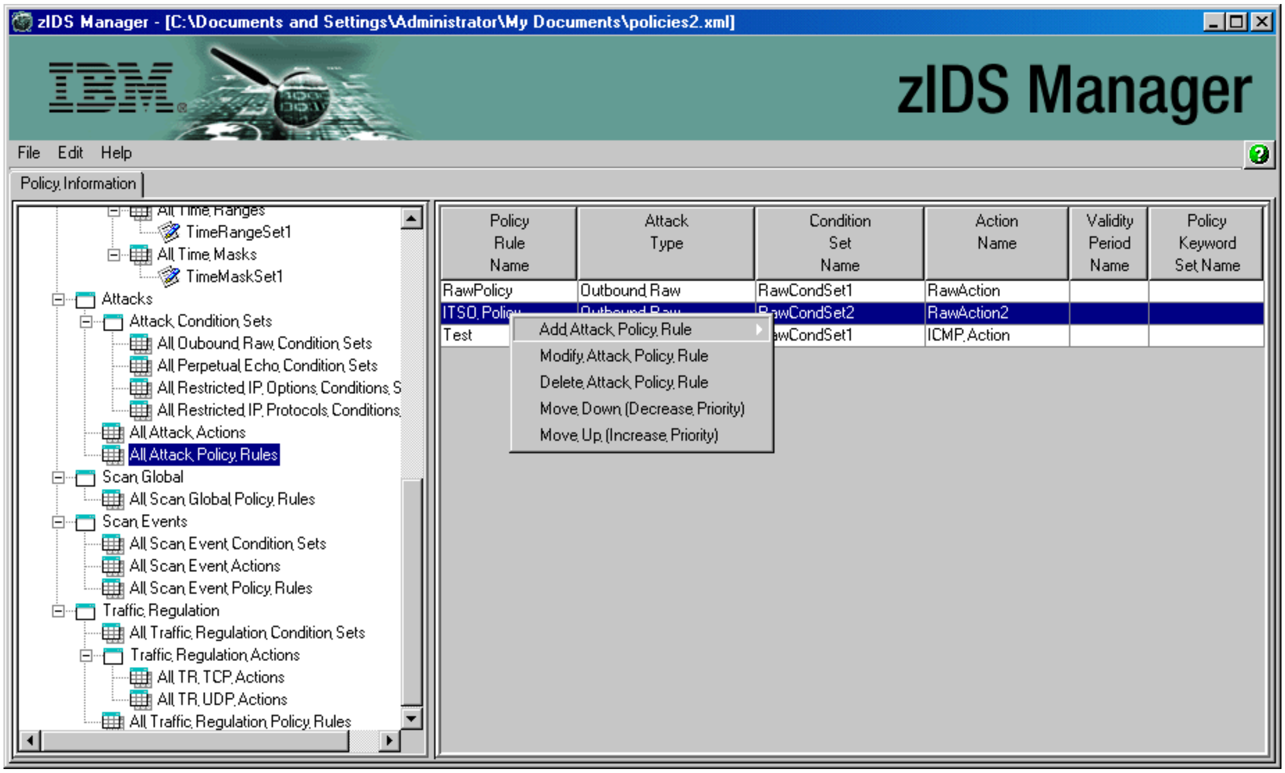
Figure 10-33 TR TCP policy

10.5 Policy priorities

Policies consist of several related objects. The main object is the policy rule. A policy rule object refers to one policy condition and one policy action with optional validity periods and policy keywords. Policy objects are analogous to an IF/THEN statement in a program. For example:

```
IF condition THEN action
```

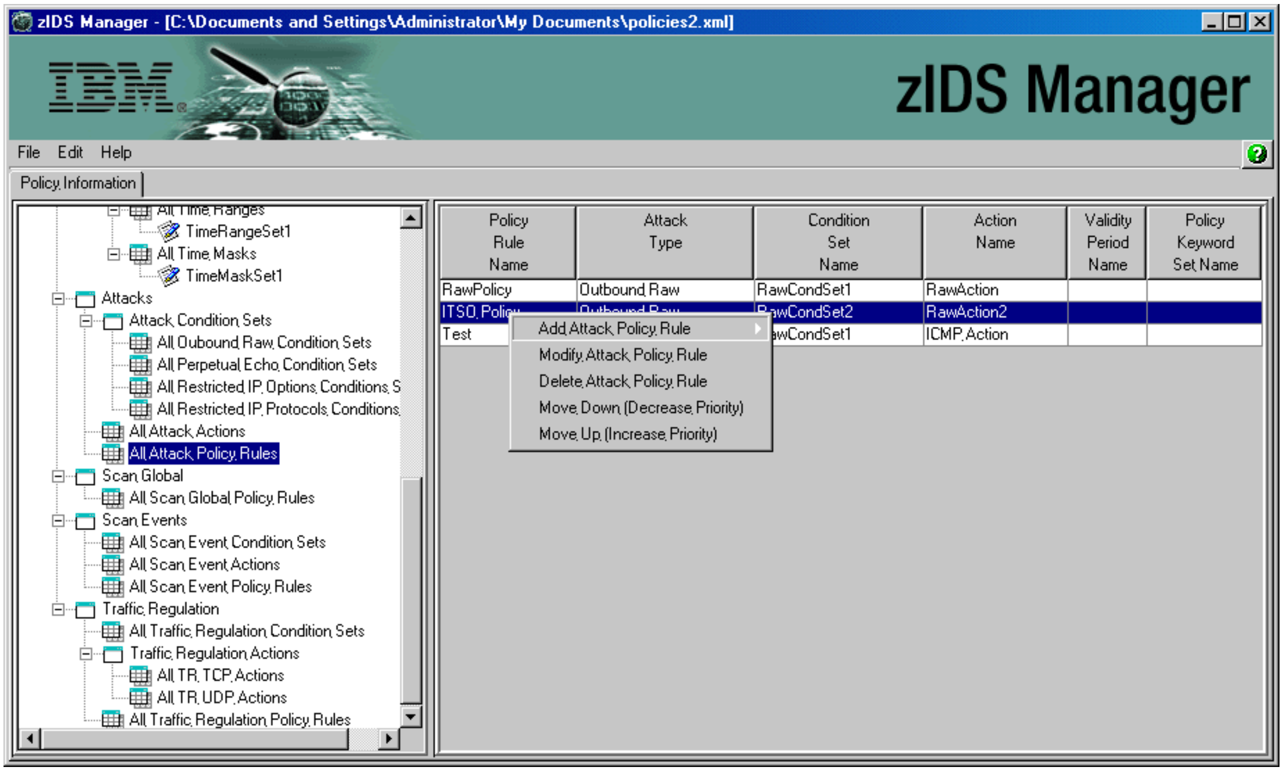
In other words, when the set of conditions referred to by a policy rule is TRUE, then the policy actions associated with the policy rule are executed. Only one policy rule and associated action can be applied to a particular packet. The priortization of the policy can be seen when you add a policy and receive the **Above Selection/Below Selection** pop-up menu, as shown in Figure 10-34 on page 228.


zIDS Manager - [C:\Documents and Settings\Administrator\My Documents\policies2.xml]
zIDS Manager
File
Edit
Help
Policy Information
All Time Ranges
TimeRangeSet1
All Time Masks
TimeMaskSet1
Attacks
Attack Condition Sets
All Oubound Raw Condition Sets
All Perpetual Echo Condition Sets
All Restricted IP Options Conditions S
All Restricted IP Protocols Conditions
All Attack Actions
All Attack Policy Rules
Scan Global
All Scan Global Policy Rules
Scan Events
All Scan Event Condition Sets
All Scan Event Actions
All Scan Event Policy Rules
Traffic Regulation
All Traffic Regulation Condition Sets
Traffic Regulation Actions
All TR TCP Actions
All TR UDP Actions
All Traffic Regulation Policy Rules
Add Attack Policy Rule
Modify Attack Policy Rule
Delete Attack Policy Rule
Move Down (Decrease Priority)
Move Up (Increase Priority)

Policy Rule Name	Attack Type	Condition Set Name	Action Name	Validity Period Name	Policy Keyword Set Name
RawPolicy	Outbound Raw	RawCondSet1	RawAction		
ITSO Policy	Outbound Raw	RawCondSet2	RawAction2		
Test		wCondSet1	ICMP Action		

Figure 10-34 Policy priortizaiton

The first policy in the policy rule table with a TRUE condition will be executed. Thus, the prioritization of policies must be evaluated prior to implementation. One can easily prioritize a policy by clicking a policy in the right-hand pane and when the user chooses to add a policy, they are prompted with a specification for above/below the current policy. Once a policy is placed in the table of policy of rule, the user has the ability to move the policy based on the priortization. This is done by highlighting an existing policy in the right-hand pane and then right-clicking the policy. Figure 10-35 on page 229 illustrates the available options for an existing policy in policy rule table.

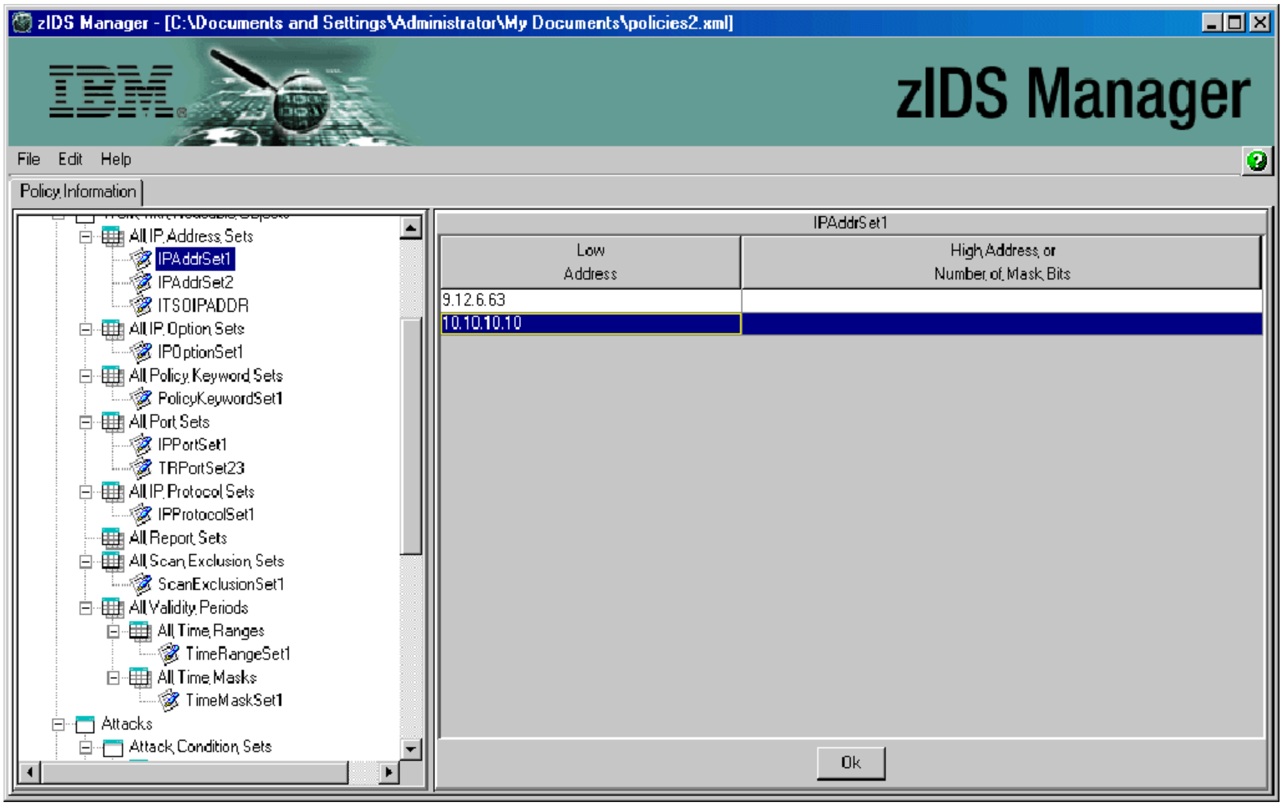

zIDS Manager - [C:\Documents and Settings\Administrator\My Documents\policies2.xml]
IBM
zIDS Manager
File
Edit
Help
Policy Information
All Time Ranges
TimeRangeSet1
All Time Masks
TimeMaskSet1
Attacks
Attack Condition Sets
All Oubound Raw Condition Sets
All Perpetual Echo Condition Sets
All Restricted IP Options Conditions S
All Restricted IP Protocols Conditions
All Attack Actions
All Attack Policy Rules
Scan Global
All Scan Global Policy Rules
Scan Events
All Scan Event Condition Sets
All Scan Event Actions
All Scan Event Policy Rules
Traffic Regulation
All Traffic Regulation Condition Sets
Traffic Regulation Actions
All TR TCP Actions
All TR UDP Actions
All Traffic Regulation Policy Rules
Add Attack Policy Rule
Modify Attack Policy Rule
Delete Attack Policy Rule
Move Down (Decrease Priority)
Move Up (Increase Priority)

Policy Rule Name	Attack Type	Condition Set Name	Action Name	Validity Period Name	Policy Keyword Set Name
RawPolicy	Outbound Raw	RawCondSet1	RawAction		
ITSO Policy	Outbound Raw	RawCondSet2	RawAction2		
Test		wCondSet1	ICMP Action		

Figure 10-35 Policy options within the policy rule table

10.5.1 Conjunctive Normal Form (CNF) policies

The zIDS Manager only supports the Conjunctive Normal Form, which means an ANDed (different condition levels) set of ORed conditions (same condition level). This ORing of the same level conditions can be seen in Figure 10-36 on page 230.



IPAddrSet1

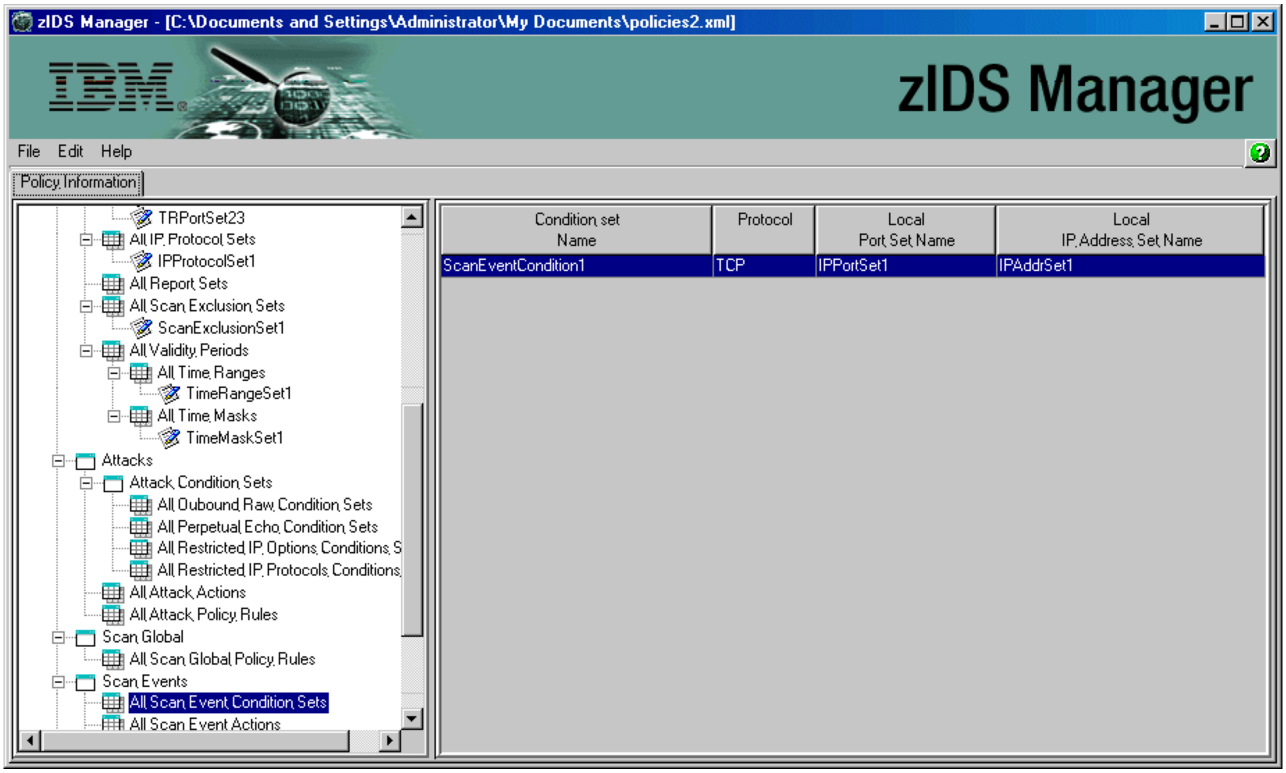
Low Address	High Address or Number of Mask Bits
9.12.6.63	
10.10.10.10	

Figure 10-36 CNF ORed condition

The information in IPAddrSet1 is all at the same level. Thus, when evaluated in a packet, this information will be ORed. This can be viewed as:

```
IF IP Address 9.12.6.63 OR IP Address 10.10.10.10 THEN Action
```

Now let us look at a Condition Set with multiple condition levels, which requires the AND function (see Figure 10-37 on page 231).



Condition set Name	Protocol	Local Port Set Name	Local IP Address Set Name
ScanEventCondition1	TCP	IPPortSet1	IPAddrSet1

Figure 10-37 zIDS ANDed condition

Using CNF, ScanEventConditionSet1 reads as:

```
IF TCP AND IPPortSet1 AND IPAddrSet1 THEN Action
```

Where:

- TCP = TCP Protocol
- IPPortSet1 = Port 20 OR Port 21
- IPAddrSet1 = 9.12.6.63 OR 10.10.10.10

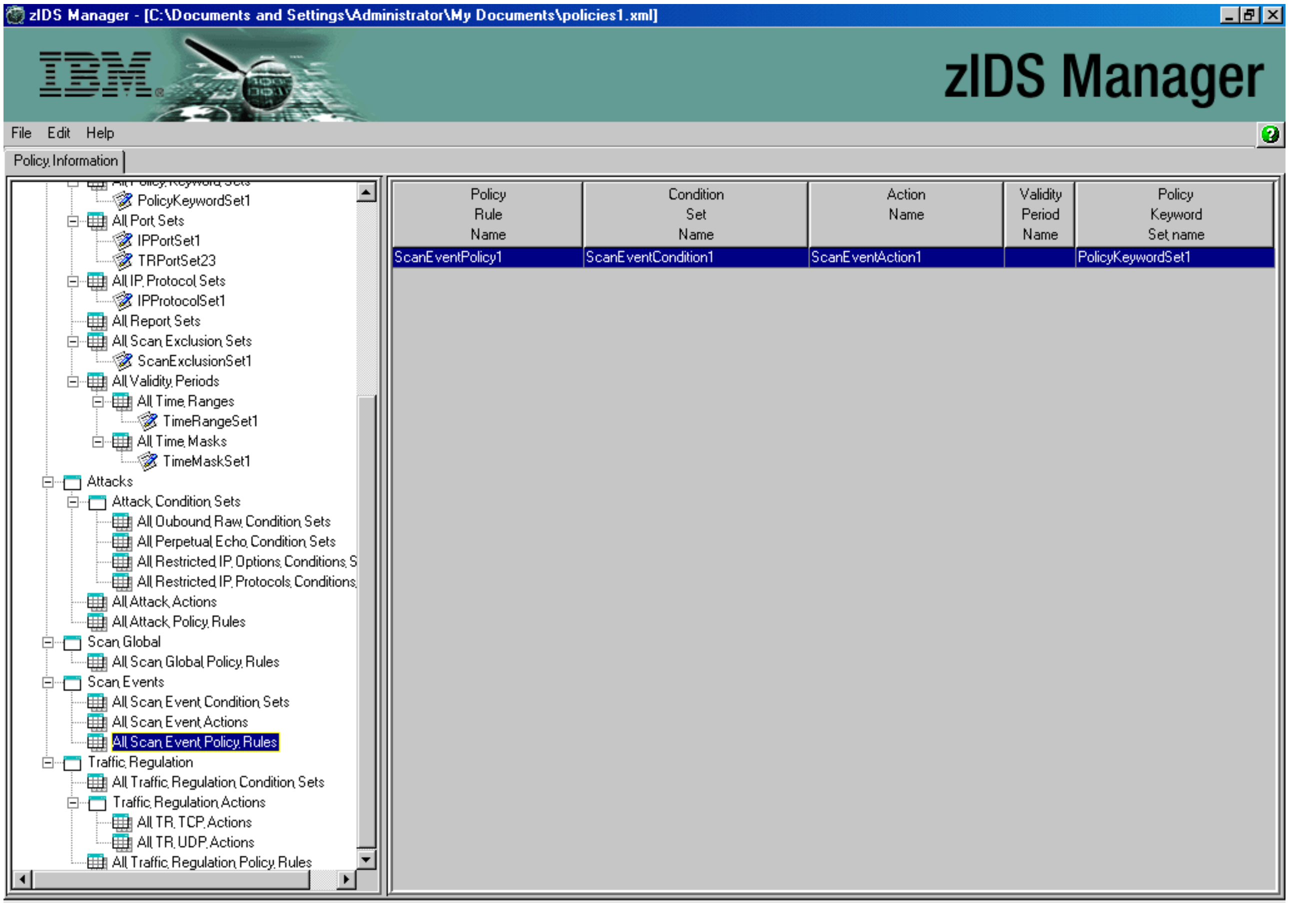
Thus making the appropriate substitutions we have:

```
If (TCP Protocol) AND (Port 20 OR Port 21) AND (9.12.6.63 OR 10.10.10.10) THEN Action
```

Now we create the action ScanEventAction1 and build the policy ScanEventPolicy1 that associates the two:

```
ScanEventPolicy1:
If (TCP Protocol) AND (Port 20 OR Port 21) AND (9.12.6.63 OR 10.10.10.10) THEN
ScanEventAction1
```

This is shown in graphical form in Figure 10-38 on page 232.



Policy Rule Name	Condition Set Name	Action Name	Validity Period Name	Policy Keyword Set name
ScanEventPolicy1	ScanEventCondition1	ScanEventAction1		PolicyKeywordSet1

Figure 10-38 zIDS graphical representation of the policy

10.6 Additional information

In this section we provide some additional information including a summary of limitations, common mistakes, and logging.

10.6.1 Limitations

The zIDS Manager TCP/IP connection with the LDAP Server cannot be a secure connection, whereas the connection between PAGENT and the LDAP Server can be secure. IBM has been notified of this limitation via the newsgroup.

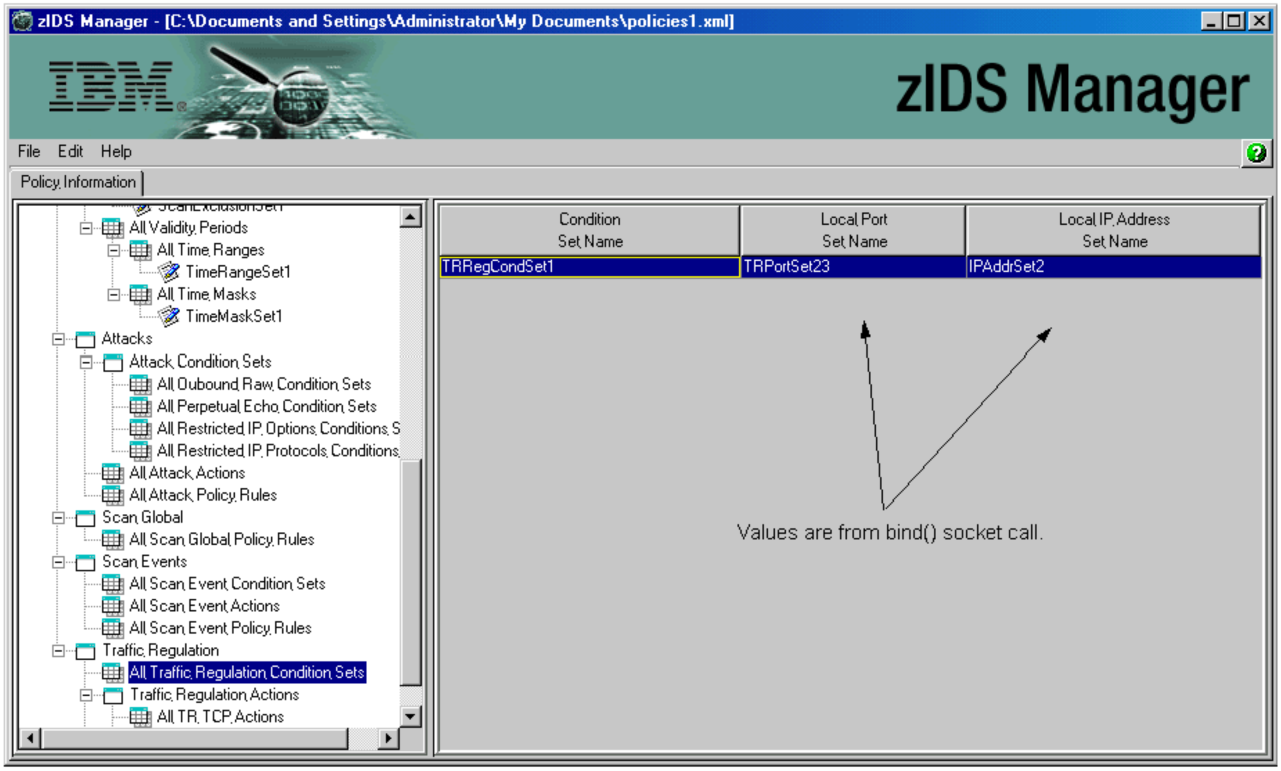
10.6.2 Common mistakes

While incorporating our policies into the zIDS Manager, we experienced problems that may be common to others.

- All Traffic Regulation Condition Sets - local values. We coded this value as the local adapter or DVIPA address we wanted to regulate the traffic on. This is not the intent of this field. The intent is discussed in “All Traffic Regulation conditions sets - local values” on page 233.
- All report sets - logging level. We did not know what to code for this field and the Help menus did not provide the information we needed to set this value.

All Traffic Regulation conditions sets - local values

Figure 10-39 is a TR Condition Set; notice the **Local Port Set Name** and **Local IP Address Set Name**. These represent reuseable objects that are based on the port and IP address that were passed in the bind() socket call. We encountered a problem when we used the IP address of the local interface we were using. We coded this in our IP address set name data set and the TR policy was not rejecting the connections. We then realized that the TN3270 listener had bound to port 23 with the IP address of INADDRANY (0.0.0.0). Once the change was made to the object set, the policy was effective.


zIDS Manager - [C:\Documents and Settings\Administrator\My Documents\policies1.xml]
IBM
zIDS Manager
File
Edit
Help
Policy Information
ScanExclusionSet1
All Validity Periods
All Time Ranges
TimeRangeSet1
All Time Masks
TimeMaskSet1
Attacks
Attack Condition Sets
All Oubound Raw Condition Sets
All Perpetual Echo Condition Sets
All Restricted IP Options Conditions S
All Restricted IP Protocols Conditions
All Attack Actions
All Attack Policy Rules
Scan Global
All Scan Global Policy Rules
Scan Events
All Scan Event Condition Sets
All Scan Event Actions
All Scan Event Policy Rules
Traffic Regulation
All Traffic Regulation Condition Sets
Traffic Regulation Actions
All TR TCP Actions

Condition Set Name	Local Port Set Name	Local IP Address Set Name
TRRegCondSet1	TRPortSet23	IPAddrSet2

Values are from bind() socket call.

Figure 10-39 TR Condition Set values

All report sets - logging level

The **Logging Level** field represents the syslogd value for the *priority code*. Syslogd receives the information from TRMD and processes the information based on the *facility name,* which is daemon, and the priority code specified on the call. The help screen refers you to *OS/390 V2R10.0 C/C++ Run-Time Library Reference,* SC28-1663, however, no numeric values are supplied for the priority code argument. The definitions for priority code values are as follows:

LOG_EMERG A Panic condition. This is normally broadcast to all processes.

LOG_ALERT A condition that should be corrected immediately, such as a corrupted system database.

LOG_CRIT Critical conditions, such as hard device errors.

LOG_ERR Errors.

LOG_WARNING Warning messages.

LOG_NOTICE Conditions that are not error conditions, but that may require special handling.

LOG_INFO Informational messages.

LOG_DEBUG Messages that contain information normally of use only when debugging a program.

Table 10-1 lists the corresponding numeric values needed to place in the **Logging Level** field.

Table 10-1 Syslogd priority table

Priority	**Value**
LOG_EMERG	0
LOG_ALERT	1
LOG_CRIT	2
LOG_ERR	3
LOG_WARNING	4
LOG_NOTICE	5
LOG_INFO	6
LOG_DEBUG	7

Note: The STATISTICS reporting option uses priority code, 6, LOG_INFO.

This field is not the same as the LogLevel options coded in the PAGENT configuration file.

11

Operating IDS in a real-time environment

This chapter presents the operation of IDS in a real-time environment. This includes starting the Policy Agent and all pertinent applications required to successfully incorporate IDS into your environment, reviewing existing policies and various commands to monitor the system. Several scenarios to include a scanner and an attack, and finally, modifying policies based on collected information.

This chapter presents a base scenario that the system programmer can use to meet his or her specific needs with respect to intrusion detection.

This chapter contains the following sections:

11.1 Overview

The goal of IDS is to both prevent future and stop present intrusions. In order to accomplish this goal, the policies must be functioning as designed and modified as needed in a real-time environment. This chapter is written to help the network administrator verify the following:

- Are the policies active?
- Is the expected traffic mapping to the correct policies?
- Are the IDS policy functions working correctly?
- Does anything need to be tuned?

The following sections provide key commands and displays for the network administrator to use when operating IDS in a real-time environment. The scenario that follows assumes the policies are loaded in the LDAP Server running local on the z/OS system. The following is a synopsis of the policies:

```
Pre-Built Policies were built and loaded in the LDAP server using the zIDS Manager
software:
Attack Policy Name: Raw Policy
   Attack Type: OUTBOUND RAW,
   Condition: Restricted Protocol=6
   Action:   Max Events Message=5,Type Action=Statistics,Exception Statistics,Log,Limit,
             Notification=syslogd detail,limit,Statistics Interval=60,
             Logging Level=0,Record Size=200
Scan Global Policy: ScanGlobalPolicy1
   Fast Scan Interval=1,Fast Scan Threshold=5,Slow Scan Interval=120,SlowScan Threshold=10,
   Action: Max Events Message=5,Type Actions=Exception Statistics,Statistics,Log,Limit,
          Notification=syslogd detail,console,Statistics Interval=60,
          Logging Level=0,Trace Data=Record Size,Trace Record Size=200
   PolicyKeyword: Policy1
Scan Event Policy: ScanEventPolicy1
   Condition:Protocol=TCP,Ports=1-1025,IPAddress=9.12.6.63
   Action:Sensitivity=Low, Scan Exclusion=9.19.22.22,Ports 1-123
   Policy Keyword: Policy1
Traffic Regulation Policy: TRPolicyTelnet
   Condition: Port=23,IPAddress=0.0.0.0 (INADDRANY)
   Action:Type Action=limit,log,Exception Statistics, Notification=Console,Syslog
Detail,Interval=60,Total Connections=50,Percentage=100 percent,Port=Port_Instance
          Logging Level=0, Trace Data=Header
   Policy Keyword: Policy1
```

The LDIF file generated from these policies is available in Appendix B, “LDIF example” on page 269.

As per Figure 11-1 on page 237, you can see the names of the started tasks that will repeatedly appear in our display commands on MVS image SC64. Also, the diagram depicts the network administrator’s TSO session, assumed to be a TN3270 connection from a remote personal computer. This is where the MVS console and USS shell commands are issued. Also on the PC is the zIDS Manager software. The zIDS Manager provides a user-friendly GUI for policy configuration. For more information on the zIDS Manager see Chapter 10, “IDS configuration using zIDS Manager” on page 199. Thus, the network administrator has the ability to display IDS information, make the necessary changes to his policies, update the LDAP Server, and refresh PAGENT, all from his personal computer.

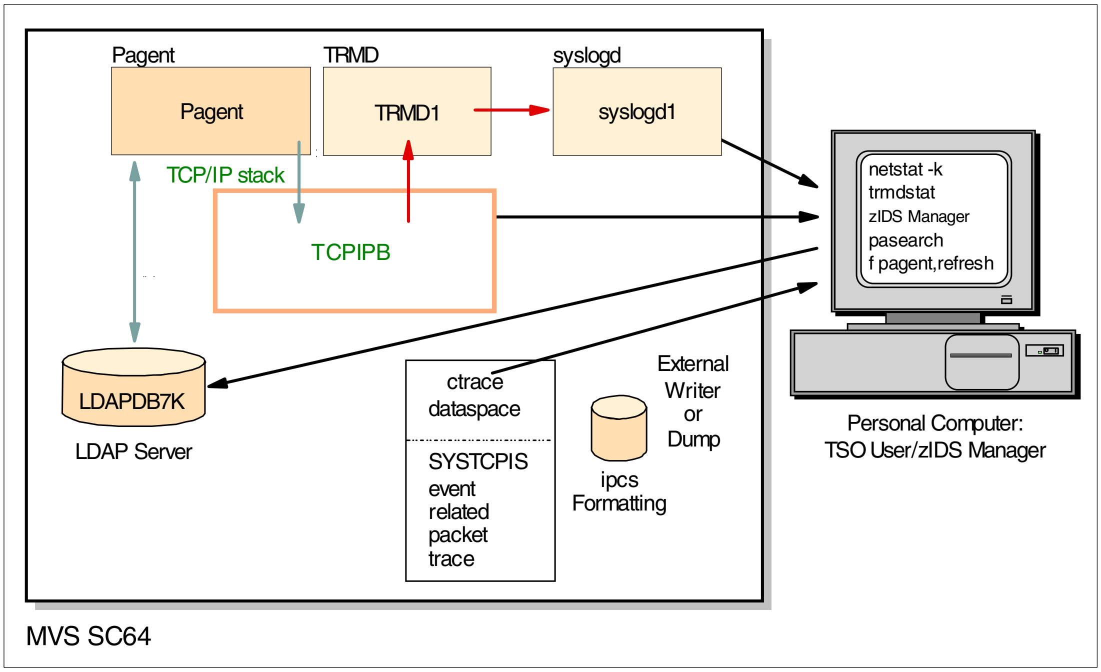


Figure 11-1 IDS real-time scenario started task names and communication flow

11.2 Starting the IDS applications

In the following sections we discuss starting the IDS applications.

11.2.1 PAGENT activation

First we must start the PAGENT address space. We will start PAGENT as a started task. See Chapter 4, “Working with z/OS Policy Agent” on page 71, for detailed information. From the MVS console, issue the following command:

```
S PAGENT
```

You will see the following messages in the job log if PAGENT was successfully started:

```
EZZ8431I PAGENT STARTING
EZZ8771I PAGENT CONFIG POLICY PROCESSING COMPLETE FOR TCPIPB
EZZ8432I PAGENT INITIALIZATION COMPLETE
```

Note: IDS policies must be stored in the LDAP Server. Thus, the PAGENT configuration file must have, and successfully execute, the ReadFromDirectory statement to incorporate the IDS policies into the TCP/IP stack.

11.2.2 PAGENT client and LDAP Server connectivity

The connection between Pagnet and the LDAP Server is a TCP connection and remains active for as long as both applications are active. The connection can be viewed through a **`netstat conn`** command. The **`netstat conn`** we issued was via the MVS console. Alternatively the command could have been issued in the USS shell or through the TSO command prompt. The command and output are shown in Figure 11-2 on page 238.

```
D TCPIP,TCPIPB,N,CONN

EZZ2500I NETSTAT CS V1R2 TCPIPB 335
USER ID  CONN     LOCAL SOCKET           FOREIGN SOCKET         STATE
LDAPDB7K 00001ACE 9.12.6.63..3389        9.12.6.63..1078        ESTBLSH
LDAPDB7K 00001AC8 0.0.0.0..3389          0.0.0.0..0             LISTEN
PAGENT   00001ACB 9.12.6.63..1078        9.12.6.63..3389        ESTBLSH
```

Figure 11-2 TCP connection between the PAGENT client and the LDAP Server

A common problem we incurred when starting the PAGENT application is when the LDAP Server is not active. In this case, you would see the following message:

```
EZZ8431I PAGENT STARTING
EZZ8432I PAGENT INITIALIZATION COMPLETE
EZZ8440I PAGENT CANNOT CONNECT TO LDAP SERVER FOR TCPIPB
EZZ8771I PAGENT CONFIG POLICY PROCESSING COMPLETE FOR TCPIPB
```

11.2.3 TRMD activation

TRMD can be started as either started task or USS process, see *z/OS V1R2 Communications Server: IP Configuration Reference*, SC31-8776, for detailed information. We will start TRMD as a started task. Issue the following MVS console command:

```
S TRMD
```

A successful initialization of TRMD will render the message shown in Figure 11-3.

```
EZZ8495I TRMD STARTED
EZZ8500I TRMD INITIALIZATION COMPLETE
$HASP395 TRMD     ENDED
```

Figure 11-3 Successful initialization of TRMD

Notice that we receive the message `TRMD ENDED` in Figure 11-3. This is normal. TRMD becomes a daemon process by forking another copy of the program, fork(), with the job name TRMD1. TRMD will then end and the new MVS job name of interest is TRMD1. This process is known as *demonizing.*

Note: If TRMD was started through the USS shell then the demonized job name would not be TRMD1. The forked process is the TSO user ID followed by a suffix.

11.3 Policy review

There are essential policy questions the network administrator must be able to answer at all times. They are:

- What IDS policies are defined in PAGENT?
- Which IDS policies are active?
- Has there been activity on any of the policies?
- Is there a reporting tool to produce a summary of the information?

These questions are answered through the **`pasearch`**, **`netstat ids`**, and **`trmdstat`** commands.

11.3.1 Terminology

- Active Policy

  A policy is active when it has been loaded from PAGENT into the TCP/IP stack.

- Active Policy Priority

  Policy selection is based on the priority of the policy.

  – TR Policy Priority: There can be only one active policy per socket (that is, local port, local IP address, and protocol) and it is based on the priority of the TR policy. This means that any inbound packet is checked against the highest priority policy for the socket it matches. If the condition is FALSE, the next highest priority policy becomes the active policy for the socket and its condition is checked, and so on.

  – Scan Global Policy Priority: The highest priority scan global policy is used for mapping.

  – Scan Event Policy Priority: There can be only one active policy per socket (that is, local port, local IP address, and protocol) and it is based on the priority of the scan event. This means that any inbound packet is checked against the highest priority policy for the socket it matches. If the condition is FALSE, the next highest priority policy becomes the active policy for the socket and its condition is checked, and so on.

  – Attack Policy Priority: The highest priority attack policy per attack type is used for each of the eight attack types.

- Mapping

  Mapping is the process of the TCP/IP stack correlating a packet to a policy. Mapping only occurs when a packet can be associated with a policy.

11.3.2 pasearch

The z/OS UNIX **pasearch** command is a shell-based command used to query information from PAGENT. The command is issued from the UNIX System Services shell. The -i option only displays the IDS policies, thus reducing the amount of information. The -A option only displays active policies. This is especially useful if you have time-based policies. For more information on the **pasearch** command and its options, see *z/OS V1R2 Communications Server: IP System Administrator's Commands*, SC31-8781. If you are running in a common inet (CINET) environment, you must ensure that you have stack affinity. From a USS shell prompt type:

```
export
```

The display output includes your USS environment variables. One of the variables is _BPXK_SETIBMOPT_TRANSPORT. Ensure the correct TCP/IP stack appears in this environment variable. If it does not, you can easily change it by issuing:

```
export _BPXK_SETIBMOPT_TRANSPORT= xxxxx
```

Where xxxxx= TCP/IP stack.

Now type the following:

```
pasearch -i
```

The following output is an example of the output for the RawPolicy, which is an attack-based policy. Section 11.4, "Scenarios" on page 244, will have the complete output when the command is issued:

```
MVS TCP/IP pasearch CS V1R2           TCP/IP Image:     TCPIPB
  Date:             05/15/2002        Time:  16:47:12
```

```
policyRule:             RawPolicy 1
  Version:              3                 Status:            Active 2
  Distinguish Name:  3 cn=RawPolicy,cn=attackRules,ou=idsMgrPolicies,o=ITSO,c=US
  Weight:               355               ForLoadDist:       FALSE
  Priority:             255               Sequence Actions:  Don't Care
  ConditionListType:    CNF               No. Policy Action: 1
  IdsType:              policyIdsAttack
  policyAction:         RawAction
   ActionType:          IDS               Action Sequence:   0
  Time Periods:
   Day of Month Mask:
   First to Last:       1111111111111111111111111111111
   Last to First:       1111111111111111111111111111111
   Month of Year Mask:  111111111111
   Day of Week Mask:    1111111  (Sunday - Saturday)
   Start Date Time:     None
   End Date Time:       None
   From TimeOfDay:      00:00             To TimeOfDay:      24:00
   From TimeOfDay UTC:  04:00             To TimeOfDay UTC:  04:00
   TimeZone:            Local
  Ids Condition Summary:                  NegativeIndicator: OFF
   AttackCondition:
    IdsAttackType:      outbound_raw
    IPOptionFrom:       0                 IPOptionTo:        0
   TransportCondition:
    LocalPortFrom:      0                 LocalPortTo:       0
    RemotePortFrom:     0                 RemotePortTo:      0
    ProtocolNumFrom:    6                 ProtocolNumTo:     6
   HostCondition:
    LocalIpAddrFrom:    0.0.0.0           LocalIpAddrTo:     0.0.0.0
    RemoteIpAddrFrom:   0.0.0.0           RemoteIpAddrTo:    0.0.0.0
  Condition Work Level:      0
    Group Number:       0                 Cond Count:        1
    Ignore:             NO
  Ids Condition Work Summary:             NegativeIndicator: OFF
   AttackCondition:
    IdsAttackType:      outbound_raw
    IPOptionFrom:       0                 IPOptionTo:        0
   TransportCondition:
    LocalPortFrom:      0                 LocalPortTo:       0
    RemotePortFrom:     0                 RemotePortTo:      0
    ProtocolNumFrom:    0                 ProtocolNumTo:     0
   HostCondition:
    LocalIpAddrFrom:    0.0.0.0           LocalIpAddrTo:     0.0.0.0
    RemoteIpAddrFrom:   0.0.0.0           RemoteIpAddrTo:    0.0.0.0
  Condition Work Level:      1
    Group Number:       1                 Cond Count:        1
    Ignore:             NO
  Ids Condition Work Summary:             NegativeIndicator: OFF
   AttackCondition:
    IdsAttackType:      None
    IPOptionFrom:       0                 IPOptionTo:        0
   TransportCondition:
    LocalPortFrom:      0                 LocalPortTo:       0
    RemotePortFrom:     0                 RemotePortTo:      0
    ProtocolNumFrom:    6                 ProtocolNumTo:     6
   HostCondition:
    LocalIpAddrFrom:    0.0.0.0           LocalIpAddrTo:     0.0.0.0
    RemoteIpAddrFrom:   0.0.0.0           RemoteIpAddrTo:    0.0.0.0
  Ids Action:           RawAction
    Version:            3                 Status:            Active
    IdsType:            policyIdsAttack
    Distinguish Name:
cn=RawAction,cn=attackActions,cn=policyRepository,ou=idsMgrPolicies,o=ITSO,c=US
    Notification Attributes
     Notification:      Syslog SyslogDetail Console
     TraceData:         RecordSize
     TypeActions:       Statistics Log Limit ExceptStats
     StatInterval:      60                LoggingLevel:      0
     TraceRecordSize:   200
    Attack Actions Attributes
     MaxEventMessage:   5
```

The following can be seen:

1. Raw policy.
2. The policy is status, which is active.
3. The Distinguished Name information.

The policy output is in hierarchical order based on the policy rule. Notice that the policy is in active state on stack TCPIPB and policies

11.3.3 netstat ids

The netstat ids command provides IDS mapping information from the TCP/IP stack. The command can be issued either from the USS shell, TSO commands, or the console. In our example, the command was issued as a console command.

The syntax of the command from the console is,

```
D TCPIP,tcpip procname,N,IDS SUM
or
D TCPIP,tcpip procname,N,IDS PROTOCOL protocol
```

We have first issued the command with no prior mapping occurrences. The output is:

```
D TCPIP,TCPIPB,N,IDS PROTOCOL TCP
EZZ2500I NETSTAT CS V1R2 TCPIPB 897
INTRUSION DETECTION SERVICES SUMMARY:
SCAN DETECTION:
  GLOBRULENAME: SCANGLOBALPOLICY1
  ICMPRULENAME: *NONE*
  TOTDETECTED:  0           DETCURRPLC: 0
  DETCURRINT:   0           INTERVAL:   30
   SRCIPSTRKD:   0          STRGLEV:    00000
ATTACK DETECTION:
   MALFORMED PACKETS
     PLCRULENAME: *NONE*
     TOTDETECTED: 0          DETCURRPLC: 0
     DETCURRINT:  0          INTERVAL:   0
   OUTBOUND RAW RESTRICTIONS
     PLCRULENAME: RAWPOLICY
     TOTDETECTED: 0          DETCURRPLC: 0
     DETCURRINT:  0          INTERVAL:   60
   RESTRICTED PROTOCOLS
     PLCRULENAME: *NONE*
     TOTDETECTED: 0          DETCURRPLC: 0
     DETCURRINT:  0          INTERVAL:   0
  RESTRICTED IP OPTIONS
```

```
   PLCRULENAME: *NONE*
   TOTDETECTED: 0           DETCURRPLC: 0
   DETCURRINT:  0           INTERVAL:   0
 ICMP REDIRECT RESTRICTIONS
   PLCRULENAME: *NONE*
   TOTDETECTED: 0           DETCURRPLC: 0
   DETCURRINT:  0           INTERVAL:   0
 IP FRAGMENT RESTRICTIONS
   PLCRULENAME: *NONE*
   TOTDETECTED: 0           DETCURRPLC: 0
   DETCURRINT:  0           INTERVAL:   0
 UDP PERPETUAL ECHO
   PLCRULENAME: *NONE*
    TOTDETECTED: 0           DETCURRPLC: 0
    DETCURRINT:  0           INTERVAL:   0
  FLOODS
    PLCRULENAME: *NONE*
    TOTDETECTED: 0           DETCURRPLC: 0
    DETCURRINT:  0           INTERVAL:   0
TRAFFIC REGULATION:
  TCP
    CONNREJECTED: 0           PLCACTIVE: Y
  UDP
    PCKDISCARDED: 0           PLCACTIVE: N
INTRUSION DETECTION SERVICES TCP PORT LIST:
INTRUSION DETECTION SERVICES UDP PORT LIST:
```

We will see these values change in 11.4, “Scenarios” on page 244, when attacks and scans are detected.

11.3.4 trmdstat

trmdstat is a utility program that runs from the USS shell. trmdstat reads a log file, analyzes the log records generated by TRMD, and provides summary or detailed reports based on the options specified.

The following can be requested:

- Overall summary of logged connection events
- IDS summary of logged events
- Reports of logged connection events
- Reports of logged intrusions defined in the attack policy
- Reports of logged intrusions defined in the TCP policy
- Reports of logged intrusions defined in the UDP policy
- Reports of statistics events

Figure 11-4 on page 243 show the summary output for the **trmdstat** command:

```
trmdstat -I log.file
```

```
trmd -I /u/syslogd/sc64.syslog
trmdstat for z/OS CS V1R2          Thu May 23 14:04:59 2002

Stack Name           : ALL
Log Time Interval    :  -
Stack Time Interval  :  -
TRM Records Scanned  : 0
Port Range           : ALL

TCP - Traffic Regulation
------------------------------------------------
Connections would have been refused :          0
Connections refused                 :          0

Constrained entry logged            :          0
Constrained exit logged             :          0
Constrained entry                   :          0
Constrained exit                    :          0

QOS exceptions logged               :          0
QOS exceptions made                 :          0

UDP - Traffic Regulation
------------------------------------------------
Constrained entry logged            :          0
Constrained exit logged             :          0
Constrained entry                   :          0
Constrained exit                    :          0

SCAN Detection
------------------------------------------------
Threshold exceeded                  :          0
Detection delayed                   :          0
Storage constrained entry           :          0
Storage constrained exit            :          0

ATTACK Detection
------------------------------------------------
Packet would have been discarded    :          0
Packet discarded                    :          0

FLOOD Detection
------------------------------------------------
Accept queue expanded               :          0
SYN flood start                     :          0
SYN flood end                       :          0
```

Figure 11-4   trmdstat IDS overall summary report

Refer to *z/OS V1R2 Communications Server: IP User's Guide and Commands,* SC31-8780, for more information on the **trmdstat** command and samples of the reports generated by **trmdstat**. Sample outputs are also shown in 11.4.1, "Scenario 1 - Scanner" on page 250, and 11.4.2, "Scenario 2 - Traffic Regulation" on page 253.

11.4 Scenarios

Two scenarios have been generated to demonstrate the messages, logging, and denial of service produced when the defined policies are mapped correctly. The two scenarios are:

- Scenario 1: There is a scanner from IP address 9.42.103.86. Based on our scan event policies, ScanGlobalPolicy1 and ScanEventPolicy1, a fast scan has been detected.
- Scenario 2: Traffic Regulation policy TRPolicyTelnet is mapped and enforces the denial of service.

The two intrusion programs that were written for scenario 1 and scenario 2 are located in Appendix C, “IDS scan and Traffic Regulation programs” on page 283.

As described in 11.3.2, “pasearch” on page 239, the IDS **pasearch** command is:

```
pasearch -i
```

This command produces the same results in either scenario 1 or scenario 2. The reason is that from the PAGENT perspective, the scenarios (attacks, scans, TR), are independent of the policies that are already in place. Thus, the pasearch -i results are shown below:

```
MVS TCP/IP pasearch CS V1R2             TCP/IP Image:      TCPIPB
  Date:                 05/15/2002       Time:  16:47:12

policyRule:             RawPolicy
  Version:              3                 Status:            Active
  Distinguish Name:     cn=RawPolicy,cn=attackRules,ou=idsMgrPolicies,o=ITSO,c=US
  Weight:               355               ForLoadDist:       FALSE
  Priority:             255               Sequence Actions:  Don't Care
  ConditionListType:    CNF               No. Policy Action: 1
  IdsType:              policyIdsAttack
  policyAction:         RawAction
   ActionType:          IDS               Action Sequence:   0
  Time Periods:
   Day of Month Mask:
   First to Last:       1111111111111111111111111111111
   Last to First:       1111111111111111111111111111111
   Month of Year Mask:  111111111111
   Day of Week Mask:    1111111  (Sunday - Saturday)
   Start Date Time:     None
   End Date Time:       None
   From TimeOfDay:      00:00             To TimeOfDay:      24:00
   From TimeOfDay UTC:  04:00             To TimeOfDay UTC:  04:00
   TimeZone:            Local
  Ids Condition Summary:                  NegativeIndicator: OFF
   AttackCondition:
    IdsAttackType:      outbound_raw
    IPOptionFrom:       0                 IPOptionTo:        0
   TransportCondition:
    LocalPortFrom:      0                 LocalPortTo:       0
    RemotePortFrom:     0                 RemotePortTo:      0
    ProtocolNumFrom:    6                 ProtocolNumTo:     6
   HostCondition:
    LocalIpAddrFrom:    0.0.0.0           LocalIpAddrTo:     0.0.0.0
    RemoteIpAddrFrom:   0.0.0.0           RemoteIpAddrTo:    0.0.0.0
  Condition Work Level:      0
    Group Number:       0                 Cond Count:        1
    Ignore:             NO
  Ids Condition Work Summary:             NegativeIndicator: OFF
   AttackCondition:
    IdsAttackType:      outbound_raw
    IPOptionFrom:       0                 IPOptionTo:        0
   TransportCondition:
    LocalPortFrom:      0                 LocalPortTo:       0
    RemotePortFrom:     0                 RemotePortTo:      0
    ProtocolNumFrom:    0                 ProtocolNumTo:     0
   HostCondition:
    LocalIpAddrFrom:    0.0.0.0           LocalIpAddrTo:     0.0.0.0
    RemoteIpAddrFrom:   0.0.0.0           RemoteIpAddrTo:    0.0.0.0
  Condition Work Level:      1
    Group Number:       1                 Cond Count:        1
    Ignore:             NO
  Ids Condition Work Summary:             NegativeIndicator: OFF
   AttackCondition:
    IdsAttackType:      None
    IPOptionFrom:       0                 IPOptionTo:        0
   TransportCondition:
    LocalPortFrom:      0                 LocalPortTo:       0
    RemotePortFrom:     0                 RemotePortTo:      0
    ProtocolNumFrom:    6                 ProtocolNumTo:     6
   HostCondition:
    LocalIpAddrFrom:    0.0.0.0           LocalIpAddrTo:     0.0.0.0
    RemoteIpAddrFrom:   0.0.0.0           RemoteIpAddrTo:    0.0.0.0

  Ids Action:           RawAction
    Version:            3                 Status:            Active
    IdsType:            policyIdsAttack
    Distinguish Name:
cn=RawAction,cn=attackActions,cn=policyRepository,ou=idsMgrPolicies,o=ITSO,c=US
    Notification Attributes
     Notification:      Syslog SyslogDetail Console
     TraceData:         RecordSize
     TypeActions:       Statistics Log Limit ExceptStats
     StatInterval:      60                LoggingLevel:      0
     TraceRecordSize:   200
    Attack Actions Attributes
     MaxEventMessage:   5

policyRule:             ScanEventPolicy1
  Version:              3                 Status:            Active
  Distinguish Name:     cn=ScanEventPolicy1,cn=scanEventRules,ou=idsMgrPolicies,o=ITSO,c=US
  Weight:               355               ForLoadDist:       FALSE
  Priority:             255               Sequence Actions:  Don't Care
  ConditionListType:    CNF               No. Policy Action: 1
  IdsType:              policyIdsScanEvent
  policyAction:         ScanEventAction1
   ActionType:          IDS               Action Sequence:   0
  Time Periods:
   Day of Month Mask:
   First to Last:       1111111111111111111111111111111
   Last to First:       1111111111111111111111111111111
   Month of Year Mask:  111111111111
   Day of Week Mask:    1111111  (Sunday - Saturday)
   Start Date Time:     None
   End Date Time:       None
   From TimeOfDay:      00:00             To TimeOfDay:      24:00
   From TimeOfDay UTC:  04:00             To TimeOfDay UTC:  04:00
   TimeZone:            Local
  Ids Condition Summary:                  NegativeIndicator: OFF
   AttackCondition:
```

```
  IdsAttackType:      None
  IPOptionFrom:       0                 IPOptionTo:        0
 TransportCondition:
  LocalPortFrom:      1                 LocalPortTo:       1025
  RemotePortFrom:     0                 RemotePortTo:      0
  ProtocolNumFrom:    0                 ProtocolNumTo:     0
 HostCondition:
  LocalIpAddrFrom:    9.12.6.63         LocalIpAddrTo:     9.12.6.63
  RemoteIpAddrFrom:   0.0.0.0           RemoteIpAddrTo:    0.0.0.0
Condition Work Level:      0
  Group Number:       0                 Cond Count:        1
  Ignore:             NO
Ids Condition Work Summary:             NegativeIndicator: OFF
 AttackCondition:
  IdsAttackType:      None
  IPOptionFrom:       0                 IPOptionTo:        0
 TransportCondition:
  LocalPortFrom:      0                 LocalPortTo:       0
  RemotePortFrom:     0                 RemotePortTo:      0
  ProtocolNumFrom:    0                 ProtocolNumTo:     0
 HostCondition:
  LocalIpAddrFrom:    0.0.0.0           LocalIpAddrTo:     0.0.0.0
  RemoteIpAddrFrom:   0.0.0.0           RemoteIpAddrTo:    0.0.0.0
Condition Work Level:      1
  Group Number:       1                 Cond Count:        1
  Ignore:             NO
Ids Condition Work Summary:             NegativeIndicator: OFF
 AttackCondition:
  IdsAttackType:      None
  IPOptionFrom:       0                 IPOptionTo:        0
 TransportCondition:
  LocalPortFrom:      0                 LocalPortTo:       0
  RemotePortFrom:     0                 RemotePortTo:      0
  ProtocolNumFrom:    6                 ProtocolNumTo:     6
 HostCondition:
  LocalIpAddrFrom:    0.0.0.0           LocalIpAddrTo:     0.0.0.0
  RemoteIpAddrFrom:   0.0.0.0           RemoteIpAddrTo:    0.0.0.0
Condition Work Level:      2
  Group Number:       2                 Cond Count:        1
  Ignore:             NO
Ids Condition Work Summary:             NegativeIndicator: OFF
 AttackCondition:
  IdsAttackType:      None
  IPOptionFrom:       0                 IPOptionTo:        0
 TransportCondition:
  LocalPortFrom:      1                 LocalPortTo:       1025
  RemotePortFrom:     0                 RemotePortTo:      0
  ProtocolNumFrom:    0                 ProtocolNumTo:     0
 HostCondition:
  LocalIpAddrFrom:    0.0.0.0           LocalIpAddrTo:     0.0.0.0
  RemoteIpAddrFrom:   0.0.0.0           RemoteIpAddrTo:    0.0.0.0
Condition Work Level:      3
  Group Number:       5                 Cond Count:        1
  Ignore:             NO
Ids Condition Work Summary:             NegativeIndicator: OFF
 AttackCondition:
  IdsAttackType:      None
  IPOptionFrom:       0                 IPOptionTo:        0
 TransportCondition:
  LocalPortFrom:      0                 LocalPortTo:       0
```

```
    RemotePortFrom:     0                 RemotePortTo:      0
    ProtocolNumFrom:    0                 ProtocolNumTo:     0
   HostCondition:
    LocalIpAddrFrom:    9.12.6.63         LocalIpAddrTo:     9.12.6.63
    RemoteIpAddrFrom:   0.0.0.0           RemoteIpAddrTo:    0.0.0.0

  Ids Action:           ScanEventAction1
    Version:            3                 Status:            Active
    IdsType:            policyIdsScanEvent
    Distinguish Name:
cn=ScanEventAction1,cn=scanEventActions,cn=policyRepository,ou=idsMgrPolicies,o=ITSO,c=US
    Notification Attributes
     Notification:      None
     TraceData:         Header
     TypeActions:       0
     StatInterval:      60                LoggingLevel:      0
     TraceRecordSize:   100
    Scan Event Attributes
     Sensitivity:       Low
     Scan Exclusion:    0
      ScanExclAddress:  9.19.22.22        ScanExclMask       FFFFFFFF
      ExclFromPort:     1                 ExclToPort:        123

policyRule:             ScanGlobalPolicy1
  Version:              3                 Status:            Active
  Distinguish Name:
cn=ScanGlobalPolicy1,cn=scanGlobalRules,ou=idsMgrPolicies,o=ITSO,c=US
  Weight:               355               ForLoadDist:       FALSE
  Priority:             255               Sequence Actions:  Don't Care
  ConditionListType:    CNF               No. Policy Action: 1
  IdsType:              policyIdsScanGlobal
  policyAction:         gScanActAssoc255
   ActionType:          IDS               Action Sequence:   0
  Time Periods:
   Day of Month Mask:
   First to Last:       1111111111111111111111111111111
   Last to First:       1111111111111111111111111111111
   Month of Year Mask:  111111111111
   Day of Week Mask:    1111111  (Sunday - Saturday)
   Start Date Time:     None
   End Date Time:       None
   From TimeOfDay:      00:00             To TimeOfDay:      24:00
   From TimeOfDay UTC:  04:00             To TimeOfDay UTC:  04:00
   TimeZone:            Local
  Ids Condition Summary:                  NegativeIndicator: OFF
   AttackCondition:
    IdsAttackType:      None
    IPOptionFrom:       0                 IPOptionTo:        0
   TransportCondition:
    LocalPortFrom:      0                 LocalPortTo:       0
    RemotePortFrom:     0                 RemotePortTo:      0
    ProtocolNumFrom:    0                 ProtocolNumTo:     0
   HostCondition:
    LocalIpAddrFrom:    0.0.0.0           LocalIpAddrTo:     0.0.0.0
    RemoteIpAddrFrom:   0.0.0.0           RemoteIpAddrTo:    0.0.0.0

  Ids Action:           gScanActAssoc255
    Version:            3                 Status:            Active
    IdsType:            policyIdsScanGlobal
    Distinguish Name:
cn=gScanActAssoc255,cn=ScanGlobalPolicy1,cn=scanGlobalRules,ou=idsMgrPolicies,o=ITSO,c=US
    Notification Attributes
     Notification:      Syslog SyslogDetail Console
     TraceData:         RecordSize
     TypeActions:       Statistics Log Limit ExceptStats
     StatInterval:      60                LoggingLevel:      0
     TraceRecordSize:   200
    Scan Global Attributes
     FastScanInterval:  1                 FastScanThreshold: 5
     SlowScanInterval:  120               SlowScanThreshold: 10

policyRule:             TRPolicyTelnet
  Version:              3                 Status:            Active
  Distinguish Name:     cn=TRPolicyTelnet,cn=trRules,ou=idsMgrPolicies,o=ITSO,c=US
  Weight:               355               ForLoadDist:       FALSE
  Priority:             255               Sequence Actions:  Don't Care
  ConditionListType:    CNF               No. Policy Action: 1
  IdsType:              policyIdsTR
  policyAction:         TRTCPAction1
   ActionType:          IDS               Action Sequence:   0
  Time Periods:
   Day of Month Mask:
   First to Last:       1111111111111111111111111111111
   Last to First:       1111111111111111111111111111111
   Month of Year Mask:  111111111111
   Day of Week Mask:    1111111  (Sunday - Saturday)
   Start Date Time:     None
   End Date Time:       None
   From TimeOfDay:      00:00             To TimeOfDay:      24:00
   From TimeOfDay UTC:  04:00             To TimeOfDay UTC:  04:00
   TimeZone:            Local
  Ids Condition Summary:                  NegativeIndicator: OFF
   AttackCondition:
    IdsAttackType:      None
    IPOptionFrom:       0                 IPOptionTo:        0
   TransportCondition:
    LocalPortFrom:      23                LocalPortTo:       23
    RemotePortFrom:     0                 RemotePortTo:      0
    ProtocolNumFrom:    0                 ProtocolNumTo:     0
   HostCondition:
    LocalIpAddrFrom:    0.0.0.0           LocalIpAddrTo:     0.0.0.0
    RemoteIpAddrFrom:   0.0.0.0           RemoteIpAddrTo:    0.0.0.0
  Condition Work Level:      0
    Group Number:       0                 Cond Count:        1
    Ignore:             NO
  Ids Condition Work Summary:             NegativeIndicator: OFF
   AttackCondition:
    IdsAttackType:      None
    IPOptionFrom:       0                 IPOptionTo:        0
   TransportCondition:
    LocalPortFrom:      0                 LocalPortTo:       0
    RemotePortFrom:     0                 RemotePortTo:      0
    ProtocolNumFrom:    0                 ProtocolNumTo:     0
   HostCondition:
    LocalIpAddrFrom:    0.0.0.0           LocalIpAddrTo:     0.0.0.0
    RemoteIpAddrFrom:   0.0.0.0           RemoteIpAddrTo:    0.0.0.0
  Condition Work Level:      1
    Group Number:       1                 Cond Count:        1
    Ignore:             NO
  Ids Condition Work Summary:             NegativeIndicator: OFF
   AttackCondition:
    IdsAttackType:      None
    IPOptionFrom:       0                 IPOptionTo:        0
   TransportCondition:
    LocalPortFrom:      0                 LocalPortTo:       0
    RemotePortFrom:     0                 RemotePortTo:      0
    ProtocolNumFrom:    6                 ProtocolNumTo:     6
   HostCondition:
    LocalIpAddrFrom:    0.0.0.0           LocalIpAddrTo:     0.0.0.0
    RemoteIpAddrFrom:   0.0.0.0           RemoteIpAddrTo:    0.0.0.0
  Condition Work Level:      2
    Group Number:       2                 Cond Count:        1
    Ignore:             NO
  Ids Condition Work Summary:             NegativeIndicator: OFF
   AttackCondition:
    IdsAttackType:      None
    IPOptionFrom:       0                 IPOptionTo:        0
   TransportCondition:
    LocalPortFrom:      23                LocalPortTo:       23
    RemotePortFrom:     0                 RemotePortTo:      0
    ProtocolNumFrom:    0                 ProtocolNumTo:     0
   HostCondition:
    LocalIpAddrFrom:    0.0.0.0           LocalIpAddrTo:     0.0.0.0
    RemoteIpAddrFrom:   0.0.0.0           RemoteIpAddrTo:    0.0.0.0
  Condition Work Level:      3
    Group Number:       5                 Cond Count:        1
    Ignore:             NO
  Ids Condition Work Summary:             NegativeIndicator: OFF
   AttackCondition:
    IdsAttackType:      None
    IPOptionFrom:       0                 IPOptionTo:        0
   TransportCondition:
    LocalPortFrom:      0                 LocalPortTo:       0
    RemotePortFrom:     0                 RemotePortTo:      0
    ProtocolNumFrom:    0                 ProtocolNumTo:     0
   HostCondition:
    LocalIpAddrFrom:    0.0.0.0           LocalIpAddrTo:     0.0.0.0
    RemoteIpAddrFrom:   0.0.0.0           RemoteIpAddrTo:    0.0.0.0

  Ids Action:           TRTCPAction1
    Version:            3                 Status:            Active
    IdsType:            policyIdsTR
    Distinguish Name:
cn=TRTCPAction1,cn=trTCPActions,cn=policyRepository,ou=idsMgrPolicies,o=ITSO,c=US
    Notification Attributes
     Notification:      Console
     TraceData:         Header
     TypeActions:       Log Limit
     StatInterval:      60                LoggingLevel:      0
     TraceRecordSize:   100
    Traffic Regulation Attributes
     TcpTotConnections: 8                 TcpPercentage:     100
     TcpLimitScope:     PortInstance
     UDPQueueSize:      Very Long
```

11.4.1 Scenario 1 - Scanner

The ScanGlobalPolicy1 policy we defined in the beginning of the chapter has the following action/notification parameters:

- Notification: syslogd detail, console
- Action: Exception Statistics, Statistics, Limit, Log
- Trace Data=record Size,Trace Record Size=200

> **Note:** Limit and Log keywords in the Action statement are used in TR policies only.

Trace data was specified, therefore, ctrace entries are written to component SYSTCPIS. The SYSTCPIS ctrace formatter is based on the SYSTCPDA formatter (and in fact shares many of the data structures and format routines) and includes the reports for the SYSTCPDA formatter (packet trace formatter). See *z/OS V1R2.0 CS: IP Diagnosis* for information on starting and selecting the proper options for your IDS trace writer.

The following sections show information related to this scenario.

Console message

The following steps show information related to Figure 11-5.

1. A Fast Scan was detected.
2. The correlator is Correlates an IDS message on the system console or in syslogd with packets in the IDS trace (SYSTCPIS).
3. The ProbeID identifies the code location where the intrusion was detected.

The IDS rule displays the Policy name.

```
EZZ8761I IDS EVENT DETECTED 760
EZZ8762I EVENT TYPE: FAST SCAN DETECTED  1
EZZ8763I CORRELATOR 4 2 - PROBEID 0300FFF1 3
EZZ8764I SOURCE IP ADDRESS 9.42.103.86 - PORT 0
EZZ8766I IDS RULE ScanGlobalPolicy1  4
EZZ8767I IDS ACTION gScanActAssoc255
```

Figure 11-5 Scanner console message

syslogd

The following steps show information related to Figure 11-6 on page 251.

1. This is the correlator value, which allows us to connect the syslogd messages with the console and IDS trace messages.
2. The event count represents the number of events counted during that interval. Every fifth event count causes a new message to be issued. In this case we had a total event count of 10. The reason is that once the event count reached the slow scan threshold, IDS knows that a scan has been detected and it is not necessary to continue counting.

```
May 16 18:01:02 wtsc64oe TRMD.TCPIPBÝ83951887¨: EZZ8643I TRMD SCAN threshold
exceeded:05/16/2002
18:00:38.63,sipaddr=9.42.103.86,scantype=F,pthreshold=5,pinterval=1,vs=10,ps=0,norm=0,
correlator=4 1
May 16 18:01:02 wtsc64oe TRMD.TCPIPBÝ83951887¨: EZZ8644I TRMD SCAN detail:05/16/2002
18:00:38.63,sipaddr=9.42.103.86,correlator=4,event count=4 2,event list:6,9.12.6.63,5,V;
6,9.12.6.63,6,V; 6,9.12.6.63,7,V; 6,9.12.6.63,8,V

May 16 18:01:02 wtsc64oe TRMD.TCPIPBÝ83951887¨: EZZ8644I TRMD SCAN detail:05/16/2002
18:00:38.63,sipaddr=9.42.103.86,correlator=4,event count=4,2event list:6,9.12.6.63,9,V;
6,9.12.6.63,10,V; 6,9.12.6.63,1,V; 6,9.12.6.63,2,V

May 16 18:01:02 wtsc64oe TRMD.TCPIPBÝ83951887¨: EZZ8644I TRMD SCAN detail:05/16/2002
18:00:38.63,sipaddr=9.42.103.86,correlator=4,event count=2 2,event list:6,9.12.6.63,3,V;
6,9.12.6.63,4,V; ;
```

Figure 11-6 Syslogd detail entries from scanner

trmdstat

We can also run a trmdstat report for the scan summary. The command is:

```
trmdstat -N /tmp/trmd.log
```

This produces the report shown in Figure 11-7.

```
trmdstat for z/OS CS V1R2          Thu May 16 16:06:58 2002

Stack Name           : ALL
Log Time Interval    : May 16 18:01:02  - May 16 19:35:32
Stack Time Interval  : May 16 18:00:38  - May 16 19:35:10
TRM Records Scanned  : 11
Port Range           : ALL

                             SCAN  Summary

  IP Address           Scans                 Suspicion Level
                   Fast      Slow       Very      Possibly     Normal
--------------- ---------- ---------- ---------- ---------- ----------
9.42.103.86             1          0       10          0          0


TRMD Ended                   : May 16 19:40:02
```

Figure 11-7 trmdstat scan command

netstat ids

The **netstat** command also references the scanner (see Figure 11-10 on page 254).

```
D TCPIP,TCPIPB,N,IDS
EZZ2500I NETSTAT CS V1R2 TCPIPB 482
INTRUSION DETECTION SERVICES SUMMARY:
SCAN DETECTION:
  GLOBRULENAME: SCANGLOBALPOLICY1
  ICMPRULENAME: *NONE*
  TOTDETECTED:  1 1         DETCURRPLC: 0
  DETCURRINT:   0           INTERVAL:   30
  SRCIPSTRKD:   0           STRGLEV:    00000
ATTACK DETECTION:
  MALFORMED PACKETS
    PLCRULENAME: *NONE*
    TOTDETECTED: 0           DETCURRPLC: 0
    DETCURRINT:  0           INTERVAL:   0
  OUTBOUND RAW RESTRICTIONS
    PLCRULENAME: RAWPOLICY
    TOTDETECTED: 0           DETCURRPLC: 0
    DETCURRINT:  0           INTERVAL:   60
  RESTRICTED PROTOCOLS
    PLCRULENAME: *NONE*
    TOTDETECTED: 0           DETCURRPLC: 0
    DETCURRINT:  0           INTERVAL:   0
  RESTRICTED IP OPTIONS
    PLCRULENAME: *NONE*
    TOTDETECTED: 0           DETCURRPLC: 0
    DETCURRINT:  0           INTERVAL:   0
  ICMP REDIRECT RESTRICTIONS
    PLCRULENAME: *NONE*
    TOTDETECTED: 0           DETCURRPLC: 0
    DETCURRINT:  0           INTERVAL:   0
  IP FRAGMENT RESTRICTIONS
      PLCRULENAME: *NONE*
    TOTDETECTED: 0           DETCURRPLC: 0
    DETCURRINT:  0           INTERVAL:   0
  UDP PERPETUAL ECHO
    PLCRULENAME: *NONE*
    TOTDETECTED: 0           DETCURRPLC: 0
    DETCURRINT:  0           INTERVAL:   0
  FLOODS
    PLCRULENAME: *NONE*
    TOTDETECTED: 0           DETCURRPLC: 0
    DETCURRINT:  0           INTERVAL:   0
TRAFFIC REGULATION:
  TCP
    CONNREJECTED: 0           PLCACTIVE: Y
  UDP
    PCKDISCARDED: 0           PLCACTIVE: N
INTRUSION DETECTION SERVICES TCP PORT LIST:
TCPLISTENINGSOCKET: 0.0.0.0..23
```

Figure 11-8 netstat ids after scanner

The scanner was detected and the information is saved in the TCP/IP stack.

11.4.2 Scenario 2 - Traffic Regulation

The TRPolicyTelnet policy we defined in the beginning of the chapter has the following action/notification parameters:

- Notification: Console, Syslogd Detail
- Action: Limit, Log, Exception Statistics
- Total Connections: 50, Percentage:100

The Traffic Regulation causes the following to occur,

Console message

The following steps show information related to Figure 11-9.

1. A TR TCP connection limit was reached.
2. The correlator is Correlates an IDS message on the system console or in syslogd with packets in the IDS trace (SYSTCPIS).
3. The ProbeID identifies the code location where the intrusion was detected.
4. The IDS rule displays the policy name.

```
EZZ8761I IDS EVENT DETECTED 557
EZZ8762I EVENT TYPE: TCP SOURCE IP CONNECTION LIMIT REACHED 1
EZZ8763I CORRELATOR 6 - PROBEID 01004044       2 3
EZZ8764I SOURCE IP ADDRESS 9.42.103.86 - PORT 0
EZZ8765I DESTINATION IP ADDRESS 0.0.0.0 - PORT 23
EZZ8766I IDS RULE TRPolicyTelnet      4
EZZ8767I IDS ACTION TRTCPAction1
```

Figure 11-9 Traffic Regulation console messages

syslogd

Figure 11-10 on page 254The correlator and Probeid match the output from the console. There will be a TR log record entry in syslogd for every rejected connection.

```
May 16 20:57:19 wtsc64oe TRMD.TCPIPBÝ65609¨: EZZ9324I TRMD TCP connection
refused:05/16/2002
20:56:56.42,lhost=0.0.0.0,port=23,host=9.42.103.86,available=25,total=50,percent=100,cor
relator=6,probeid=01004044,host_current=25 1
May 16 20:57:19 wtsc64oe TRMD.TCPIPBÝ65609¨: EZZ9324I TRMD TCP connection
refused:05/16/2002
20:56:56.48,lhost=0.0.0.0,port=23,host=9.42.103.86,available=25,total=50,percent=100,cor
relator=6,probeid=01004044,host_current=25
May 16 20:57:19 wtsc64oe TRMD.TCPIPBÝ65609¨: EZZ9324I TRMD TCP connection
refused:05/16/2002
20:56:56.53,lhost=0.0.0.0,port=23,host=9.42.103.86,available=25,total=50,percent=100,cor
relator=6,probeid=01004044,host_current=25
May 16 20:57:19 wtsc64oe TRMD.TCPIPBÝ65609¨: EZZ9324I TRMD TCP connection
refused:05/16/2002
20:56:56.57,lhost=0.0.0.0,port=23,host=9.42.103.86,available=25,total=50,percent=100,cor
relator=6,probeid=01004044,host_current=25
May 16 20:57:19 wtsc64oe TRMD.TCPIPBÝ65609¨: EZZ9324I TRMD TCP connection
refused:05/16/2002
20:56:56.63,lhost=0.0.0.0,port=23,host=9.42.103.86,available=25,total=50,percent=100,cor
relator=6,probeid=01004044,host_current=25
May 16 20:57:19 wtsc64oe TRMD.TCPIPBÝ65609¨: EZZ9324I TRMD TCP connection
refused:05/16/2002
20:56:56.68,lhost=0.0.0.0,port=23,host=9.42.103.86,available=25,total=50,percent=100,cor
relator=6,probeid=01004044,host_current=25
.
.
.
```

Figure 11-10 syslogd Logging for TR Policy

netstat ids

The following shows information related to Figure 11-11 on page 255.

1 The total number of connections rejected during the current statistical interval.

2 The socket which IDS has mapped connections.

3 The name of the Traffic Regulation Policy associated with the socket.

4 The total number of connections rejected over the lifetime of the policy.

```
D TCPIP,TCPIPB,N,IDS
EZZ2500I NETSTAT CS V1R2 TCPIPB 482
INTRUSION DETECTION SERVICES SUMMARY:
SCAN DETECTION:
  GLOBRULENAME: SCANGLOBALPOLICY1
  ICMPRULENAME: *NONE*
  TOTDETECTED:  1          DETCURRPLC: 0
  DETCURRINT:   0           INTERVAL:   30
  SRCIPSTRKD:   0           STRGLEV:    00000
ATTACK DETECTION:
  MALFORMED PACKETS
    PLCRULENAME: *NONE*
    TOTDETECTED: 0           DETCURRPLC: 0
    DETCURRINT:  0           INTERVAL:   0
  OUTBOUND RAW RESTRICTIONS
    PLCRULENAME: RAWPOLICY
    TOTDETECTED: 0           DETCURRPLC: 0
    DETCURRINT:  0           INTERVAL:   60
  RESTRICTED PROTOCOLS
    PLCRULENAME: *NONE*
    TOTDETECTED: 0           DETCURRPLC: 0
    DETCURRINT:  0           INTERVAL:   0
  RESTRICTED IP OPTIONS
    PLCRULENAME: *NONE*
    TOTDETECTED: 0           DETCURRPLC: 0
    DETCURRINT:  0           INTERVAL:   0
  ICMP REDIRECT RESTRICTIONS
    PLCRULENAME: *NONE*
    TOTDETECTED: 0           DETCURRPLC: 0
    DETCURRINT:  0           INTERVAL:   0
  IP FRAGMENT RESTRICTIONS
      PLCRULENAME: *NONE*
    TOTDETECTED: 0           DETCURRPLC: 0
    DETCURRINT:  0           INTERVAL:   0
  UDP PERPETUAL ECHO
    PLCRULENAME: *NONE*
    TOTDETECTED: 0           DETCURRPLC: 0
    DETCURRINT:  0           INTERVAL:   0
  FLOODS
    PLCRULENAME: *NONE*
    TOTDETECTED: 0           DETCURRPLC: 0
    DETCURRINT:  0           INTERVAL:   0
TRAFFIC REGULATION:
  TCP
    CONNREJECTED: 94 1        PLCACTIVE: Y
  UDP
    PCKDISCARDED: 0           PLCACTIVE: N
INTRUSION DETECTION SERVICES TCP PORT LIST:
TCPLISTENINGSOCKET: 0.0.0.0..23  2
  SCSTAT: C  SCRULENAME: *NONE*
  TRSTAT: C  TRRULENAME: TRPOLICYTELNET   3
  TRPORTINST: Y  TRCORR: 0          MXAPP: 0          MXHST: 94  4
  SYNFLOOD:   N
INTRUSION DETECTION SERVICES UDP PORT LIST:
1 OF 1 RECORDS DISPLAYED
```

Figure 11-11 netstat ids for TR policy

trmdstat

The following **trmdstat** command was issued to obtain the report (as seen in Table 11-12),

```
trmdstat -p 23 trmd.log
```

```
Appendix 15-13, “trmdstat” on page 86
 trmdstat for z/OS CS V1R2          Thu May 16 17:25:42 2002

 Stack Name           : ALL
 Log Time Interval    : May 16 20:57:19  - May 16 20:57:19
 Stack Time Interval  : May 16 20:56:56  - May 16 20:57:01
 TRM Records Scanned  : 96
 Port Range           : 23 - 23

 TCP - Traffic Regulation
 ------------------------------------------------
 Connections would have been refused :         0
 Connections refused                 :        94

 Constrained entry logged            :         0
 Constrained exit logged             :         0
 Constrained entry                   :         0
 Constrained exit                    :         0

 QOS exceptions logged               :         0
 QOS exceptions made                 :         0



 TRMD Ended                   : May 16 21:07:49
```

Figure 11-12 trmdstat for TR policy

11.5 Policy modification

Policies are reviewed and modified on a regular basis. The modification of the policies consists of the following steps:

1. Modifying the policy. This is done either through the use of the zIDS Manager or manually building and loading a flat file on the machine the LDAP Server is running and issuing the necessary LDAP Server commands.
2. Refreshing PAGENT.

11.5.1 Modifying the policy

It is recommend by the ITSO Organization that the zIDS Manager be used for policy modification. After modifying the policy, click **File -> Send to LDAP**. This establishes a TCP/IP connection with the LDAP Server and the zIDS Manager. The policy modifications are then sent to the LDAP Server with the proper LDAP Server commands.

Note: The communication with the zIDS manager and the LDAP Server is one-way communication. The zIDS Manager sends information to the LDAP Server.

11.5.2 Refreshing PAGENT

Once the LDAP Server is updated, the next step is to refresh PAGENT. There are three ways to refresh PAGENT.

1. Issue a modify refresh command.
2. Stop/start PAGENT.
3. Touch or modify the configuration file.

Refresh PAGENT

This is done through the following console command:

```
F REFRESH, PAGENT
```

The following console messages appear verifying the refresh was successful:

```
F PAGENT,REFRESH
EZZ8443I PAGENT MODIFY COMMAND ACCEPTED
EZZ8771I PAGENT CONFIG POLICY PROCESSING COMPLETE FOR TCPIPB
```

Stop and restart PAGENT

The stop command is issued via the console command:

```
p PAGENT
```

Subsequently, the start command is issued:

```
s PAGENT
```

The PAGENT application can also be started and stopped through USS shell commands. See 11.2.1, "PAGENT activation" on page 237, for more information and example message output.

Modification of the configuration file

The PAGENT.conf file simply needs to be open/closed for an automatic PAGENT refresh to occur. This can done through a "touch" command as well. From the USS shell, go to the directory that the PAGENT configuration file is located in and type:

```
touch PAGENT.configurationfile
where PAGENT.configuration = the name of the configuration file
```

Part 5

Appendixes

A

The Policy Agent sample configuration

This appendix contains several sample configurations of the policy agent that have been tested at ITSO Raleigh during installation and testing of CS for z/OS 1.2.

The main configuration file

```
#  File name: /etc/pagent.r2615.conf
#  IBM Communications Server for OS/390
#  SMP/E distribution path: /usr/lpp/tcpip/samples/IBM/EZAPAGCO
#
# LogLevel Statement
Loglevel 511

# ** TcpImage Statement with policy in PAGENT file
#TcpImage TCPIPA /etc/pagent.r2615a.conf  FLUSH 600
#TcpImage TCPIPB /etc/pagent.r2615b.conf  FLUSH 600
# TcpImage TCPIPC /etc/pagent.r2615c.conf  FLUSH 600

# ** TcpImage Statement with policy in LDAP server + TR policy in file
#  TcpImage TCPIPC /etc/pagent.r2615c.ldap.conf  FLUSH 600
   TcpImage TCPIPB /etc/pagent.r2615b.ldap.conf  FLUSH 600
   TcpImage TCPIPA /etc/pagent.r2615a.ldap.conf  FLUSH 600

# ** TcpImage Statement with TCPIPC policy in LDAP server & PAGENT file
# TcpImage TCPIPC /etc/pagent.r2615.ldap.conf  FLUSH 600
```

The image configuration file with the policy definitions

```
#  File name: /etc/pagent.r2615a.conf
#  IBM Communications Server for OS/390
#  SMP/E distribution path: /usr/lpp/tcpip/samples/IBM/EZAPAGCO
#
# LogLevel Statement
# Loglevel 255

# PolicyPerfMonitorForSDR Statement
#PolicyPerfMonitorForSDR  enable
#{
#  samplinginterval 60
#  LossRatioAndWeightFr 20 25
#  TimeoutRatioAndWeightFr 50 50
#  LossMaxWeightFr 95
#  TimeoutMaxWeightFr 100
#}

# SetSubnetPrioTosMask Statement
SetSubnetPrioTosMask
{
  SubnetAddr          0.0.0.0
  SubnetTosMask       11100000
  PriorityTosMapping  1 1110000
  PriorityTosMapping  1 1100000
  PriorityTosMapping  2 1010000
  PriorityTosMapping  2 1000000
  PriorityTosMapping  3 0110000
  PriorityTosMapping  3 0100000
  PriorityTosMapping  4 0010000
  PriorityTosMapping  4 0000000
}


# PolicyAction Statement
```

```
policyAction   networkcontrol
{
     policyScope    DataTraffic
     OutgoingTOS    11100000     # Precedence bits (first 3 bits)
}

policyAction   internetwork      # encapsulated network control
{
     policyScope    DataTraffic
     OutgoingTOS    11000000     #
}

policyAction   crit-realtime     # realtime data
{
     policyScope    DataTraffic
     OutgoingTOS    10100000     #
}

policyAction   interactive1
{
     policyScope    DataTraffic
     OutgoingTOS    10000000
}

policyAction   interactive2
{
     policyScope    DataTraffic
     OutgoingTOS    01100000
}

policyAction   batch1
{
     policyScope    DataTraffic
     OutgoingTOS    01000000
}

policyAction   batch2
{
     policyScope    DataTraffic
     OutgoingTOS    00100000
}

policyAction   RSVP_Action1
{
     policyScope                 RSVP
     OutgoingTOS                 01100000
     FlowServiceType             ControlledLoad
     MaxRatePerFlow              400      # 50000 bytes/second
     MaxTokenBucketPerFlow       48       # 6000  bytes
     MaxFlow                     10
}

policyAction   telnetAction
{
     policyScope    TR
     TypeActions    Statistics
     TimeInterval   15
     LoggingLevel   7
}
```

```
# PolicyAction for Sysplex Distributor
#policyAction        ftpaction
#{
#    policyScope    DataTraffic
#    MaxConnections 5         # Limit FTP concurrent connections to 50.
#    MaxRate        400       # Limit FTP connection throughput to 400
#    OutgoingTOS    01000000  # the TOS value of outgoing FTP packets.
#    outboundinterface  172.16.233.5
#    outboundinterface  172.16.233.29
#    outboundinterface  172.16.233.40
#    outboundinterface  0.0.0.0
#}

# policyRule statement
policyRule          routed           # ROUTED traffic
{
     protocolNumberRange 17
     SourcePortRange    520          # ROUTED port
     policyActionReference  networkcontrol
}

policyRule          ospf             # OSPF link advertisement traffic
{
     protocolNumberRange 89          # OSPF protocol number
     policyActionReference  networkcontrol
}

policyRule          tftpd           # TFTP traffic
{
     protocolNumberRange 17
     SourcePortRange    69          # TFTP port
     policyActionReference  batch1
}

policyRule          ftpd            # FTP traffic
{
     protocolNumberRange  6
     SourcePortRange    20 21       # Both FTP control and data ports
     policyActionReference  batch1
}

policyRule          telnetd         # telnet traffic
{
     protocolNumberRange  6
     SourcePortRange    23
     policyActionReference  interactive1
}

policyRule          web-httpd       # web traffic
{
     protocolNumberRange  6
     SourcePortRange    80
     policyActionReference  interactive2
}

policyRule          dns-udp         # domain name server udp traffic
{
     protocolNumberRange 17
     SourcePortRange    42
     policyActionReference  interactive1
```

```
}

policyRule          dns-tcp         # domain name server tcp traffic
{
     protocolNumberRange  6
     SourcePortRange    42
     policyActionReference  interactive1
}

policyRule          EE-xid
{
     protocolNumberRange 17
     SourcePortRange    12000
     policyActionReference  internetwork
}

policyRule          EE-network
{
     protocolNumberRange 17
     SourcePortRange    12001
     policyActionReference  internetwork
}

policyRule          EE-highpri
{
     protocolNumberRange 17
     SourcePortRange    12002
     policyActionReference  interactive1
}

policyRule          EE-medpri
{
     protocolNumberRange 17
     SourcePortRange    12003
     policyActionReference  batch1
}

policyRule          EE-lowpri
{
     protocolNumberRange 17
     SourcePortRange    12004
     policyActionReference  batch2
}

policyRule          RSVP_Rule1
{
     protocolNumberRange 6
     SourcePortRange    8000 8001
     policyActionReference  RSVP_Action1
}

policyRule          telnetRule
{
     DestinationPortRange  23
     policyActionReference  telnetAction
}

#Policyule for Sysplex Distributor
#policyRule           ftprule
#{
```

```
#    ProtocolNumberRange 6
#    DestinationPortRange         20 21
#    policyactionreference   ftpaction
#}
```

The image configuration file as an LDAP client

```
#  File name: /etc/pagent.r2615a.ldap.conf
#  IBM Communications Server for OS/390
#  SMP/E distribution path: /usr/lpp/tcpip/samples/IBM/EZAPAGCO
#
# LogLevel Statement
  Loglevel 511

# TcpImage Statement
# TcpImage TCPIPC /etc/pagent.r2615a.ldap.conf  FLUSH 600

# LDAP service directory
# LDAP server run on system RA03 in stack TCPIPA
# VIPA address stack TCPIPA 172.16.250.3
ReadFromDirectory
{
  LDAP_Server 172.16.250.3
  LDAP_ProtocolVersion 3
  LDAP_SchemaVersion 2
  SearchPolicyBaseDN  pg=RA2615, g=policy, o=IBM_US, c=US
# Define the TCP port number for the SSL connections
  LDAP_Port 636
# Definitions for the LDAP SSL communications
 LDAP_SSL
 {
    LDAP_SSLKeyringFile /etc/ldap/pagent.kdb
    LDAP_SSLKeyringPassword r2615
    LDAP_SSLName PAGENT_RA28
 }
}
```

The image configuration file (the policy and LDAP definitions)

```
#  File name: /etc/pagent.r2615c.ldap.conf
#  IBM Communications Server for OS/390
#  SMP/E distribution path: /usr/lpp/tcpip/samples/IBM/EZAPAGCO
#
# LogLevel Statement
# Loglevel 255

# TcpImage Statement
# TcpImage TCPIPC /etc/pagent.r2615c.ldap.conf  FLUSH 600

# LDAP service directory
# LDAP server run on system RA03 in stack TCPIPA
# VIPA address stack TCPIPA 172.16.250.3
ReadFromDirectory
{
  LDAP_Server 172.16.250.3
  LDAP_ProtocolVersion 3
  LDAP_SchemaVersion 2
```

```
# SearchPolicyBaseDN  g=policy, o=IBM_US, c=US
  SearchPolicyBaseDN  pg=RA2615, g=policy, o=IBM_US, c=US
# SearchPolicyGroupKeyWord groupA
# SearchPolicyRuleKeyWord rule1
}

# PolicyPerfMonitorForSDR Statement
PolicyPerfMonitorForSDR  enable
{
  samplinginterval 60
  LossRatioAndWeightFr 20 25
  TimeoutRatioAndWeightFr 50 50
  LossMaxWeightFr 95
  TimeoutMaxWeightFr 100
}

# policyRule statement

policyRule           ftprule
{
    ProtocolNumberRange 6
    DestinationPortRange         20 21
    policyactionreference   ftpaction
}

policyRule           SDtelnetrule
{
    ProtocolNumberRange 6
    DestinationPortRange         23
    policyactionreference   SDtelnetact
}

policyRule          telnetRule
{
     DestinationPortRange  23
     policyActionReference  telnetAction
}

# PolicyAction Statement

policyAction        ftpaction
{
    policyScope    DataTraffic
    MaxConnections 5         # Limit FTP concurrent connections to 5.
    MaxRate        400       # Limit FTP connection throughput to 400
    OutgoingTOS    01000000  # the TOS value of outgoing FTP packets.
    outboundinterface  172.16.233.5
    outboundinterface  172.16.233.29
    outboundinterface  172.16.233.40
    outboundinterface  0.0.0.0
}

policyAction        SDtelnetact
{
    policyScope    DataTraffic
    MaxConnections 20        # Limit TELNET concurrent conn. to 20.
    MaxRate        400       # Limit FTP connection throughput to 400
    OutgoingTOS    11000000  # the TOS value of outgoing FTP packets.
    outboundinterface  172.16.233.5
    outboundinterface  172.16.233.29
    outboundinterface  172.16.233.40
    outboundinterface  0.0.0.0
}

policyAction   telnetAction
{
     policyScope    TR
     TypeActions    Log Statistics
     TimeInterval   15
     LoggingLevel   7
     Percentage     10
     TotalConnections 100
}

# SetSubnetPrioTosMask Statement
SetSubnetPrioTosMask
{
  SubnetAddr          0.0.0.0
  SubnetTosMask       11100000
  PriorityTosMapping  1 1110000
  PriorityTosMapping  1 1100000
  PriorityTosMapping  2 1010000
  PriorityTosMapping  2 1000000
  PriorityTosMapping  3 0110000
  PriorityTosMapping  3 0100000
  PriorityTosMapping  4 0010000
  PriorityTosMapping  4 0000000
}
```

B

LDIF example

Here you will find the LDIF file generated from the policies used in Chapter 11, “Operating IDS in a real-time environment” on page 235. The LDIF file must load into the LDAP Server. This can either be done manually via LDAP commands on the LDAP Server, or through the use of the zIDS Manager.

Policy text

The following policy information was used to generate the LDIF file via the zIDS Manager. These polices were used in Chapter 11, “Operating IDS in a real-time environment” on page 235.

```
Pre-Built Policies were built and loaded in the LDAP server using the zIDS Manager
software:
Attack Policy Name: Raw Policy
   Attack Type: OUTBOUND RAW,
   Condition: Restricted Protocol=6
   Action:   Max Events Message=5,Type Action=Statistics,Exception Statistics,Log,Limit,
             Notification=syslogd detail,limit,Statistics Interval=60,
             Logging Level=0,Record Size=200
Scan Global Policy: ScanGlobalPolicy1
   Fast Scan Interval=1,Fast Scan Threshold=5,Slow Scan Interval=120,SlowScan Threshold=10,
   Action: Max Events Message=5,Type Actions=Exception Statistics,Statistics,Log,Limit,
          Notification=syslogd detail,console,Statistics Interval=60,
          Logging Level=0,Trace Data=Record Size,Trace Record Size=200
   PolicyKeyword: Policy1
Scan Event Policy: ScanEventPolicy1
   Condition:Protocol=TCP,Ports=1-1025,IPAddress=9.12.6.63
   Action:Sensitivity=Low, Scan Exclusion=9.19.22.22,Ports 1-123
   Policy Keyword: Policy1
Traffic Regulation Policy: TRPolicyTelnet
   Condition: Port=23,IPAddress=0.0.0.0 (INADDRANY)
   Action:Type Action=limit,log,Exception Statistics, Notification=Console,Syslog
Detail,Interval=60,Total Connections=50,Percentage=100%,Port=Port_Instance
          Logging Level=0, Trace Data=Header
Policy Keyword: Policy1
```

LDIF file

```
dn:ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:organizationalUnit
ou:idsMgrPolicies

dn:cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:policyRepository
ibm-policyRepositoryName:policyRepository

dn:cn=attackActions, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:attackActions
ibm-policyRepositoryName:attackActions

dn:cn=scanEventActions, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:scanEventActions
ibm-policyRepositoryName:scanEventActions

dn:cn=trTCPActions, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
```

```
objectclass:ibm-policyRepository
cn:trTCPActions
ibm-policyRepositoryName:trTCPActions

dn:cn=trUDPActions, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:trUDPActions
ibm-policyRepositoryName:trUDPActions

dn:cn=condTypeCondRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:condTypeCondRepository
ibm-policyRepositoryName:condTypeCondRepository

dn:cn=trAddressRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:trAddressRepository
ibm-policyRepositoryName:trAddressRepository

dn:cn=scanAddressRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:scanAddressRepository
ibm-policyRepositoryName:scanAddressRepository

dn:cn=ipOptionRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:ipOptionRepository
ibm-policyRepositoryName:ipOptionRepository

dn:cn=attackLocalPortRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:attackLocalPortRepository
ibm-policyRepositoryName:attackLocalPortRepository

dn:cn=attackRemotePortRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:attackRemotePortRepository
ibm-policyRepositoryName:attackRemotePortRepository

dn:cn=scanLocalPortRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:scanLocalPortRepository
ibm-policyRepositoryName:scanLocalPortRepository

dn:cn=trLocalPortRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:trLocalPortRepository
ibm-policyRepositoryName:trLocalPortRepository

dn:cn=ipProtocolRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
```

```
objectclass:ibm-policyRepository
cn:ipProtocolRepository
ibm-policyRepositoryName:ipProtocolRepository

dn:cn=valPeriodRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:valPeriodRepository
ibm-policyRepositoryName:valPeriodRepository

dn:cn=attackRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyGroup
cn:attackRules
ibm-policyGroupName:attackRules

dn:cn=scanEventRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyGroup
cn:scanEventRules
ibm-policyGroupName:scanEventRules

dn:cn=scanGlobalRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyGroup
cn:scanGlobalRules
ibm-policyGroupName:scanGlobalRules

dn:cn=trRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyGroup
cn:trRules
ibm-policyGroupName:trRules

dn:cn=RawAction, cn=attackActions, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyActionInstance
objectclass:ibm-policyActionAuxClass
objectclass:ibm-idsActionAuxClass
objectclass:ibm-idsAttackActionsAuxClass
objectclass:ibm-idsNotificationAuxClass
cn:RawAction
ibm-policyActionName:RawAction
ibm-idsActionType:ATTACK
ibm-idsMaxEventMessage:5
ibm-idsTypeActions:Statistics
ibm-idsTypeActions:EXCEPTSTATS
ibm-idsTypeActions:Log
ibm-idsTypeActions:Limit
ibm-idsNotification:SYSLOG
ibm-idsNotification:SYSLOGDETAIL
ibm-idsNotification:Console
ibm-idsStatInterval:60
ibm-idsLoggingLevel:0
ibm-idsTraceData:RECORDSIZE
ibm-idsTraceRecordSize:200

dn:cn=ScanEventAction1, cn=scanEventActions, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
```

```
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyActionInstance
objectclass:ibm-policyActionAuxClass
objectclass:ibm-idsActionAuxClass
objectclass:ibm-idsScanSensitivityActionAuxClass
objectclass:ibm-idsScanExclusionActionAuxClass
cn:ScanEventAction1
ibm-policyActionName:ScanEventAction1
ibm-idsActionType:SCAN_EVENT
ibm-idsSensitivity:Low
ibm-idsScanExclusion:1-9.19.22.22-32-1-123

dn:cn=TRTCPAction1, cn=trTCPActions, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyActionInstance
objectclass:ibm-policyActionAuxClass
objectclass:ibm-idsActionAuxClass
objectclass:ibm-idsNotificationAuxClass
objectclass:ibm-idsTrafficRegulationActionAuxClass
objectclass:ibm-idsTRtcpActionAuxClass
cn:TRTCPAction1
ibm-policyActionName:TRTCPAction1
ibm-idsActionType:TR
ibm-idsTRtcpTotalConnections:50
ibm-idsTRtcpPercentage:100
ibm-idsTRtcpLimitScope:Port_Instance
ibm-idsTypeActions:Limit
ibm-idsTypeActions:Log
ibm-idsTypeActions:EXCEPTSTATS
ibm-idsNotification:Console
ibm-idsNotification:SYSLOGDETAIL
ibm-idsStatInterval:60
ibm-idsLoggingLevel:0
ibm-idsTraceData:None

dn:cn=IPAddr_0, cn=scanAddressRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsHostConditionAuxClass
ibm-idsConditionType:SCAN_EVENT
cn:IPAddr_0
ibm-idsLocalHostIPAddress:3-9.12.6.63-9.12.6.63

dn:cn=IPAddr_3, cn=trAddressRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsHostConditionAuxClass
ibm-idsConditionType:TR
cn:IPAddr_3
ibm-idsLocalHostIPAddress:3-0.0.0.0-0.0.0.0
```

```
dn:cn=Option_0, cn=ipOptionRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
objectclass:ibm-idsIPAttackConditionAuxClass
ibm-idsConditionType:ATTACK
cn:Option_0
ibm-idsIPOptionRange:17

dn:cn=Option_1, cn=ipOptionRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
objectclass:ibm-idsIPAttackConditionAuxClass
ibm-idsConditionType:ATTACK
cn:Option_1
ibm-idsIPOptionRange:12

dn:cn=Option_2, cn=ipOptionRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
objectclass:ibm-idsIPAttackConditionAuxClass
ibm-idsConditionType:ATTACK
cn:Option_2
ibm-idsIPOptionRange:12

dn:cn=Port_0, cn=scanLocalPortRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsTransportConditionAuxClass
ibm-idsConditionType:SCAN_EVENT
cn:Port_0
ibm-idsLocalPortRange:1-1025

dn:cn=Port_1, cn=trLocalPortRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsTransportConditionAuxClass
ibm-idsConditionType:TR
cn:Port_1
ibm-idsLocalPortRange:23-23
```

```
dn:cn=Protocol_0, cn=ipProtocolRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsTransportConditionAuxClass
ibm-idsConditionType:ATTACK
cn:Protocol_0
ibm-idsProtocolRange:6-6

dn:cn=scan-tcp, cn=ipProtocolRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsTransportConditionAuxClass
ibm-idsConditionType:SCAN_EVENT
cn:scan-tcp
ibm-idsProtocolRange:6

dn:cn=scan-udp, cn=ipProtocolRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsTransportConditionAuxClass
ibm-idsConditionType:SCAN_EVENT
cn:scan-udp
ibm-idsProtocolRange:17

dn:cn=scan-icmp, cn=ipProtocolRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsTransportConditionAuxClass
ibm-idsConditionType:SCAN_EVENT
cn:scan-icmp
ibm-idsProtocolRange:1

dn:cn=tr-tcp, cn=ipProtocolRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsTransportConditionAuxClass
ibm-idsConditionType:TR
cn:tr-tcp
ibm-idsProtocolRange:6

dn:cn=tr-udp, cn=ipProtocolRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
```

```
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsTransportConditionAuxClass
ibm-idsConditionType:TR
cn:tr-udp
ibm-idsProtocolRange:17

dn:cn=attackMalPacketCond, cn=condTypeCondRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
cn:attackMalPacketCond
ibm-policyConditionName:attackMalPacketCond
ibm-idsConditionType:ATTACK
ibm-idsAttackType:MALFORMED_PACKET

dn:cn=attackFloodCond, cn=condTypeCondRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
cn:attackFloodCond
ibm-policyConditionName:attackFloodCond
ibm-idsConditionType:ATTACK
ibm-idsAttackType:FLOOD

dn:cn=attackOutRawCond, cn=condTypeCondRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
cn:attackOutRawCond
ibm-policyConditionName:attackOutRawCond
ibm-idsConditionType:ATTACK
ibm-idsAttackType:OUTBOUND_RAW

dn:cn=attackIcmpRedCond, cn=condTypeCondRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
cn:attackIcmpRedCond
ibm-policyConditionName:attackIcmpRedCond
ibm-idsConditionType:ATTACK
```

```
ibm-idsAttackType:ICMP_REDIRECT

dn:cn=attackPerpEchoCond, cn=condTypeCondRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
cn:attackPerpEchoCond
ibm-policyConditionName:attackPerpEchoCond
ibm-idsConditionType:ATTACK
ibm-idsAttackType:PERPETUAL_ECHO

dn:cn=attackIpFragCond, cn=condTypeCondRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
cn:attackIpFragCond
ibm-policyConditionName:attackIpFragCond
ibm-idsConditionType:ATTACK
ibm-idsAttackType:IP_FRAGMENT

dn:cn=attackRestIpOptCond, cn=condTypeCondRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
cn:attackRestIpOptCond
ibm-policyConditionName:attackRestIpOptCond
ibm-idsConditionType:ATTACK
ibm-idsAttackType:RESTRICTED_IP_OPTIONS

dn:cn=attackRestIpProtCond, cn=condTypeCondRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
cn:attackRestIpProtCond
ibm-policyConditionName:attackRestIpProtCond
ibm-idsConditionType:ATTACK
ibm-idsAttackType:RESTRICTED_IP_PROTOCOL

dn:cn=scanEventCond, cn=condTypeCondRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
```

```
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsScanEventConditionAuxClass
cn:scanEventCond
ibm-policyConditionName:scanEventCond
ibm-idsConditionType:SCAN_EVENT

dn:cn=trCond, cn=condTypeCondRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsTrafficRegulationConditionAuxClass
cn:trCond
ibm-policyConditionName:trCond
ibm-idsConditionType:TR

dn:cn=RawPolicy, cn=attackRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRule
objectclass:ibm-policySubtreesPtrAuxClass
cn:RawPolicy
ibm-policyRuleName:RawPolicy
ibm-policyRuleEnabled:1
ibm-policyRuleConditionListType:2
ibm-policyRulePriority:255

dn:cn=actAssoc255, cn=RawPolicy, cn=attackRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleActionAssociation
objectclass:ibm-policyActionAuxClass
cn:actAssoc255
ibm-policyActionName:actAssoc255
ibm-policyActionOrder:0
ibm-policyActionDN:cn=RawAction, cn=attackActions, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US

dn:cn=attack-type-255, cn=RawPolicy, cn=attackRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:attack-type-255
ibm-policyConditionName:attack-type-255
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:0
ibm-policyConditionDN:cn=attackOutRawCond, cn=condTypeCondRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US

dn:cn=attack-proto-255-0, cn=RawPolicy, cn=attackRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:attack-proto-255-0
ibm-policyConditionName:attack-proto-255-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:1
ibm-policyConditionDN:cn=Protocol_0, cn=ipProtocolRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US

dn:cn=ScanEventPolicy1, cn=scanEventRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
```

```
objectclass:ibm-policyRule
objectclass:ibm-policySubtreesPtrAuxClass
cn:ScanEventPolicy1
ibm-policyRuleName:ScanEventPolicy1
ibm-policyRuleEnabled:1
ibm-policyRuleConditionListType:2
ibm-policyRulePriority:255
ibm-policyKeyWords:Policy1

dn:cn=actAssoc255, cn=ScanEventPolicy1, cn=scanEventRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleActionAssociation
objectclass:ibm-policyActionAuxClass
cn:actAssoc255
ibm-policyActionName:actAssoc255
ibm-policyActionOrder:0
ibm-policyActionDN:cn=ScanEventAction1, cn=scanEventActions, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
ibm-policyKeyWords:Policy1

dn:cn=scanev-type-255, cn=ScanEventPolicy1, cn=scanEventRules, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:scanev-type-255
ibm-policyConditionName:scanev-type-255
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:0
ibm-policyConditionDN:cn=scanEventCond, cn=condTypeCondRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
ibm-policyKeyWords:Policy1

dn:cn=scanev-proto-255-0, cn=ScanEventPolicy1, cn=scanEventRules, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:scanev-proto-255-0
ibm-policyConditionName:scanev-proto-255-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:1
ibm-policyConditionDN:cn=scan-tcp, cn=ipProtocolRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
ibm-policyKeyWords:Policy1

dn:cn=scanev-lport-255-0, cn=ScanEventPolicy1, cn=scanEventRules, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:scanev-lport-255-0
ibm-policyConditionName:scanev-lport-255-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:2
ibm-policyConditionDN:cn=Port_0, cn=scanLocalPortRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
ibm-policyKeyWords:Policy1

dn:cn=scanev-localAddr-255-0, cn=ScanEventPolicy1, cn=scanEventRules, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
```

```
cn:scanev-localAddr-255-0
ibm-policyConditionName:scanev-localAddr-255-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:5
ibm-policyConditionDN:cn=IPAddr_0, cn=scanAddressRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
ibm-policyKeyWords:Policy1

dn:cn=ScanGlobalPolicy1, cn=scanGlobalRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRule
objectclass:ibm-policySubtreesPtrAuxClass
cn:ScanGlobalPolicy1
ibm-policyRuleName:ScanGlobalPolicy1
ibm-policyRuleEnabled:1
ibm-policyRuleConditionListType:2
ibm-policyRulePriority:255
ibm-policyKeyWords:Policy1

dn:cn=gScanassoc-255, cn=ScanGlobalPolicy1, cn=scanGlobalRules, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsScanConditionAuxClass
cn:gScanassoc-255
ibm-policyConditionName:gScanassoc-255
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:0
ibm-idsConditionType:SCAN_GLOBAL
ibm-policyKeyWords:Policy1

dn:cn=gScanActAssoc255, cn=ScanGlobalPolicy1, cn=scanGlobalRules, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleActionAssociation
objectclass:ibm-policyActionAuxClass
objectclass:ibm-idsActionAuxClass
objectclass:ibm-idsScanActionAuxClass
objectclass:ibm-idsNotificationAuxClass
cn:gScanActAssoc255
ibm-policyActionName:gScanActAssoc255
ibm-policyActionOrder:0
ibm-idsActionType:SCAN_GLOBAL
ibm-idsFSInterval:1
ibm-idsFSThreshold:5
ibm-idsSSInterval:120
ibm-idsSSThreshold:10
ibm-idsTypeActions:Statistics
ibm-idsTypeActions:EXCEPTSTATS
ibm-idsTypeActions:Log
ibm-idsTypeActions:Limit
ibm-idsNotification:SYSLOG
ibm-idsNotification:SYSLOGDETAIL
ibm-idsNotification:Console
ibm-idsStatInterval:60
ibm-idsLoggingLevel:0
ibm-idsTraceData:RECORDSIZE
ibm-idsTraceRecordSize:200
```

```
ibm-policyKeyWords:Policy1

dn:cn=TRPolicyTelnet, cn=trRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRule
objectclass:ibm-policySubtreesPtrAuxClass
cn:TRPolicyTelnet
ibm-policyRuleName:TRPolicyTelnet
ibm-policyRuleEnabled:1
ibm-policyRuleConditionListType:2
ibm-policyRulePriority:255
ibm-policyKeyWords:Policy1

dn:cn=actAssoc255, cn=TRPolicyTelnet, cn=trRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleActionAssociation
objectclass:ibm-policyActionAuxClass
cn:actAssoc255
ibm-policyActionName:actAssoc255
ibm-policyActionOrder:0
ibm-policyActionDN:cn=TRTCPAction1, cn=trTCPActions, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
ibm-policyKeyWords:Policy1

dn:cn=tr-type-255, cn=TRPolicyTelnet, cn=trRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:tr-type-255
ibm-policyConditionName:tr-type-255
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:0
ibm-policyConditionDN:cn=trCond, cn=condTypeCondRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
ibm-policyKeyWords:Policy1

dn:cn=tr-proto-255-0, cn=TRPolicyTelnet, cn=trRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:tr-proto-255-0
ibm-policyConditionName:tr-proto-255-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:1
ibm-policyConditionDN:cn=tr-tcp, cn=ipProtocolRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
ibm-policyKeyWords:Policy1

dn:cn=tr-lport-255-0, cn=TRPolicyTelnet, cn=trRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:tr-lport-255-0
ibm-policyConditionName:tr-lport-255-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:2
ibm-policyConditionDN:cn=Port_1, cn=trLocalPortRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
ibm-policyKeyWords:Policy1

dn:cn=tr-localAddr-255-0, cn=TRPolicyTelnet, cn=trRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
```

```
cn:tr-localAddr-255-0
ibm-policyConditionName:tr-localAddr-255-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:5
ibm-policyConditionDN:cn=IPAddr_3, cn=trAddressRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
ibm-policyKeyWords:Policy1
```

C

IDS scan and Traffic Regulation programs

This appendix presents the C programs that were used to simulate the Fast Scan (11.4.1, “Scenario 1 - Scanner” on page 250) and Traffic Regulation (11.4.2, “Scenario 2 - Traffic Regulation” on page 253) scenarios that were used in Chapter 11, “Operating IDS in a real-time environment” on page 235. The programs were written, compiled, and run on machine that was not in the same network as the z/OS V1R2.0 machine were PAGENT was running.

C program for fast scan scenario

```
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <sys/socket.h>
#include <netinet/in.h>

int main(int argc, char **argv)
{
  int lsock, csock, clientsock;
  int rc,len;
  int i;
  struct sockaddr_in barry;
  struct linger linger_opt;

  setenv("_EDC_ADD_ERRNO2", "1", 1);

  for (i=1;i<1000;i++)
  {
  lsock=socket(2,1,0);
  if (lsock == -1)
  {
    perror("socket() request failed");
    return 1;
  }
  setsockopt(lsock, SOL_SOCKET, SO_REUSEADDR, 1, 1);

  barry.sin_family=AF_INET;
  barry.sin_port=i;
  barry.sin_addr.s_addr = inet_addr("9.12.6.63");
  rc = connect(lsock, &barry, sizeof(barry));
  if (rc < 0)
  {
    perror("connect() failed");
  }
  else printf("connect() successful\n");
  }
  sleep(120);
  return(0);
}
```

C program for Traffic Regulation scenario

```
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <sys/socket.h>
#include <netinet/in.h>

int main(int argc, char **argv)
{
  int lsock, csock, clientsock;
  int rc,len;
  int i;
  struct sockaddr_in barry;
  struct linger linger_opt;

  setenv("_EDC_ADD_ERRNO2", "1", 1);

  for (i=1;i<120;i++)
  {
  lsock=socket(2,1,0);
  if (lsock == -1)
  {
    perror("socket() request failed");
    return 1;
  }
  setsockopt(lsock, SOL_SOCKET, SO_REUSEADDR, 1, 1);

  barry.sin_family=AF_INET;
  barry.sin_port=23;
  barry.sin_addr.s_addr = inet_addr("9.12.6.63");
  rc = connect(lsock, &barry, sizeof(barry));
  if (rc < 0)
  {
    perror("connect() failed");
  }
  else
    printf("connect() successful\n");
  }
  sleep(120);
  return(0);
}
```

Policy text

The following policy information was used to generate the LDIF file via the zIDS Manager. These polices were used in Chapter 11, “Operating IDS in a real-time environment” on page 235.

```
Pre-Built Policies were built and loaded in the LDAP server using the zIDS Manager
software:
Attack Policy Name: Raw Policy
   Attack Type: OUTBOUND RAW,
   Condition: Restricted Protocol=6
   Action:   Max Events Message=5,Type Action=Statistics,Exception Statistics,Log,Limit,
             Notification=syslogd detail,limit,Statistics Interval=60,
             Logging Level=0,Record Size=200
Scan Global Policy: ScanGlobalPolicy1
   Fast Scan Interval=1,Fast Scan Threshold=5,Slow Scan Interval=120,SlowScan Threshold=10,
   Action: Max Events Message=5,Type Actions=Exception Statistics,Statistics,Log,Limit,
           Notification=syslogd detail,console,Statistics Interval=60,
           Logging Level=0,Trace Data=Record Size,Trace Record Size=200
   PolicyKeyword: Policy1
Scan Event Policy: ScanEventPolicy1
   Condition:Protocol=TCP,Ports=1-1025,IPAddress=9.12.6.63
   Action:Sensitivity=Low, Scan Exclusion=9.19.22.22,Ports 1-123
   Policy Keyword: Policy1
Traffic Regulation Policy: TRPolicyTelnet
   Condition: Port=23,IPAddress=0.0.0.0 (INADDRANY)
   Action:Type Action=limit,log,Exception Statistics, Notification=Console,Syslog
Detail,Interval=60,Total Connections=50,Percentage=100%,Port=Port_Instance
           Logging Level=0, Trace Data=Header
Policy Keyword: Policy1
```

LDIF file

```
dn:ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:organizationalUnit
ou:idsMgrPolicies

dn:cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:policyRepository
ibm-policyRepositoryName:policyRepository

dn:cn=attackActions, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:attackActions
ibm-policyRepositoryName:attackActions

dn:cn=scanEventActions, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:scanEventActions
ibm-policyRepositoryName:scanEventActions

dn:cn=trTCPActions, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:trTCPActions
ibm-policyRepositoryName:trTCPActions

dn:cn=trUDPActions, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:trUDPActions
ibm-policyRepositoryName:trUDPActions

dn:cn=condTypeCondRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:condTypeCondRepository
ibm-policyRepositoryName:condTypeCondRepository

dn:cn=trAddressRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:trAddressRepository
ibm-policyRepositoryName:trAddressRepository

dn:cn=scanAddressRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:scanAddressRepository
ibm-policyRepositoryName:scanAddressRepository

dn:cn=ipOptionRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:ipOptionRepository
ibm-policyRepositoryName:ipOptionRepository
```

```
dn:cn=attackLocalPortRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:attackLocalPortRepository
ibm-policyRepositoryName:attackLocalPortRepository

dn:cn=attackRemotePortRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:attackRemotePortRepository
ibm-policyRepositoryName:attackRemotePortRepository

dn:cn=scanLocalPortRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:scanLocalPortRepository
ibm-policyRepositoryName:scanLocalPortRepository

dn:cn=trLocalPortRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:trLocalPortRepository
ibm-policyRepositoryName:trLocalPortRepository

dn:cn=ipProtocolRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:ipProtocolRepository
ibm-policyRepositoryName:ipProtocolRepository

dn:cn=valPeriodRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:top
objectclass:ibm-policyRepository
cn:valPeriodRepository
ibm-policyRepositoryName:valPeriodRepository

dn:cn=attackRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyGroup
cn:attackRules
ibm-policyGroupName:attackRules

dn:cn=scanEventRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyGroup
cn:scanEventRules
ibm-policyGroupName:scanEventRules

dn:cn=scanGlobalRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyGroup
cn:scanGlobalRules
ibm-policyGroupName:scanGlobalRules

dn:cn=trRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyGroup
cn:trRules
ibm-policyGroupName:trRules
```

```
dn:cn=RawAction, cn=attackActions, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyActionInstance
objectclass:ibm-policyActionAuxClass
objectclass:ibm-idsActionAuxClass
objectclass:ibm-idsAttackActionsAuxClass
objectclass:ibm-idsNotificationAuxClass
cn:RawAction
ibm-policyActionName:RawAction
ibm-idsActionType:ATTACK
ibm-idsMaxEventMessage:5
ibm-idsTypeActions:Statistics
ibm-idsTypeActions:EXCEPTSTATS
ibm-idsTypeActions:Log
ibm-idsTypeActions:Limit
ibm-idsNotification:SYSLOG
ibm-idsNotification:SYSLOGDETAIL
ibm-idsNotification:Console
ibm-idsStatInterval:60
ibm-idsLoggingLevel:0
ibm-idsTraceData:RECORDSIZE
ibm-idsTraceRecordSize:200

dn:cn=ScanEventAction1, cn=scanEventActions, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyActionInstance
objectclass:ibm-policyActionAuxClass
objectclass:ibm-idsActionAuxClass
objectclass:ibm-idsScanSensitivityActionAuxClass
objectclass:ibm-idsScanExclusionActionAuxClass
cn:ScanEventAction1
ibm-policyActionName:ScanEventAction1
ibm-idsActionType:SCAN_EVENT
ibm-idsSensitivity:Low
ibm-idsScanExclusion:1-9.19.22.22-32-1-123

dn:cn=TRTCPAction1, cn=trTCPActions, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyActionInstance
objectclass:ibm-policyActionAuxClass
objectclass:ibm-idsActionAuxClass
objectclass:ibm-idsNotificationAuxClass
objectclass:ibm-idsTrafficRegulationActionAuxClass
objectclass:ibm-idsTRtcpActionAuxClass
cn:TRTCPAction1
ibm-policyActionName:TRTCPAction1
ibm-idsActionType:TR
ibm-idsTRtcpTotalConnections:50
ibm-idsTRtcpPercentage:100
ibm-idsTRtcpLimitScope:Port_Instance
ibm-idsTypeActions:Limit
ibm-idsTypeActions:Log
ibm-idsTypeActions:EXCEPTSTATS
ibm-idsNotification:Console
ibm-idsNotification:SYSLOGDETAIL
```

```
ibm-idsStatInterval:60
ibm-idsLoggingLevel:0
ibm-idsTraceData:None

dn:cn=IPAddr_0, cn=scanAddressRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsHostConditionAuxClass
ibm-idsConditionType:SCAN_EVENT
cn:IPAddr_0
ibm-idsLocalHostIPAddress:3-9.12.6.63-9.12.6.63

dn:cn=IPAddr_3, cn=trAddressRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsHostConditionAuxClass
ibm-idsConditionType:TR
cn:IPAddr_3
ibm-idsLocalHostIPAddress:3-0.0.0.0-0.0.0.0

dn:cn=Option_0, cn=ipOptionRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
objectclass:ibm-idsIPAttackConditionAuxClass
ibm-idsConditionType:ATTACK
cn:Option_0
ibm-idsIPOptionRange:17

dn:cn=Option_1, cn=ipOptionRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
objectclass:ibm-idsIPAttackConditionAuxClass
ibm-idsConditionType:ATTACK
cn:Option_1
ibm-idsIPOptionRange:12

dn:cn=Option_2, cn=ipOptionRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
objectclass:ibm-idsIPAttackConditionAuxClass
ibm-idsConditionType:ATTACK
```

```
cn:Option_2
ibm-idsIPOptionRange:12

dn:cn=Port_0, cn=scanLocalPortRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsTransportConditionAuxClass
ibm-idsConditionType:SCAN_EVENT
cn:Port_0
ibm-idsLocalPortRange:1-1025

dn:cn=Port_1, cn=trLocalPortRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsTransportConditionAuxClass
ibm-idsConditionType:TR
cn:Port_1
ibm-idsLocalPortRange:23-23

dn:cn=Protocol_0, cn=ipProtocolRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsTransportConditionAuxClass
ibm-idsConditionType:ATTACK
cn:Protocol_0
ibm-idsProtocolRange:6-6

dn:cn=scan-tcp, cn=ipProtocolRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsTransportConditionAuxClass
ibm-idsConditionType:SCAN_EVENT
cn:scan-tcp
ibm-idsProtocolRange:6

dn:cn=scan-udp, cn=ipProtocolRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsTransportConditionAuxClass
ibm-idsConditionType:SCAN_EVENT
cn:scan-udp
```

```
ibm-idsProtocolRange:17

dn:cn=scan-icmp, cn=ipProtocolRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsTransportConditionAuxClass
ibm-idsConditionType:SCAN_EVENT
cn:scan-icmp
ibm-idsProtocolRange:1

dn:cn=tr-tcp, cn=ipProtocolRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsTransportConditionAuxClass
ibm-idsConditionType:TR
cn:tr-tcp
ibm-idsProtocolRange:6

dn:cn=tr-udp, cn=ipProtocolRepository, cn=policyRepository, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsTransportConditionAuxClass
ibm-idsConditionType:TR
cn:tr-udp
ibm-idsProtocolRange:17

dn:cn=attackMalPacketCond, cn=condTypeCondRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
cn:attackMalPacketCond
ibm-policyConditionName:attackMalPacketCond
ibm-idsConditionType:ATTACK
ibm-idsAttackType:MALFORMED_PACKET

dn:cn=attackFloodCond, cn=condTypeCondRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
cn:attackFloodCond
ibm-policyConditionName:attackFloodCond
ibm-idsConditionType:ATTACK
```

```
ibm-idsAttackType:FLOOD

dn:cn=attackOutRawCond, cn=condTypeCondRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
cn:attackOutRawCond
ibm-policyConditionName:attackOutRawCond
ibm-idsConditionType:ATTACK
ibm-idsAttackType:OUTBOUND_RAW

dn:cn=attackIcmpRedCond, cn=condTypeCondRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
cn:attackIcmpRedCond
ibm-policyConditionName:attackIcmpRedCond
ibm-idsConditionType:ATTACK
ibm-idsAttackType:ICMP_REDIRECT

dn:cn=attackPerpEchoCond, cn=condTypeCondRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
cn:attackPerpEchoCond
ibm-policyConditionName:attackPerpEchoCond
ibm-idsConditionType:ATTACK
ibm-idsAttackType:PERPETUAL_ECHO

dn:cn=attackIpFragCond, cn=condTypeCondRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
cn:attackIpFragCond
ibm-policyConditionName:attackIpFragCond
ibm-idsConditionType:ATTACK
ibm-idsAttackType:IP_FRAGMENT

dn:cn=attackRestIpOptCond, cn=condTypeCondRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
```

```
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
cn:attackRestIpOptCond
ibm-policyConditionName:attackRestIpOptCond
ibm-idsConditionType:ATTACK
ibm-idsAttackType:RESTRICTED_IP_OPTIONS

dn:cn=attackRestIpProtCond, cn=condTypeCondRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsAttackConditionAuxClass
cn:attackRestIpProtCond
ibm-policyConditionName:attackRestIpProtCond
ibm-idsConditionType:ATTACK
ibm-idsAttackType:RESTRICTED_IP_PROTOCOL

dn:cn=scanEventCond, cn=condTypeCondRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsScanEventConditionAuxClass
cn:scanEventCond
ibm-policyConditionName:scanEventCond
ibm-idsConditionType:SCAN_EVENT

dn:cn=trCond, cn=condTypeCondRepository, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyInstance
objectclass:ibm-policyConditionInstance
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsTrafficRegulationConditionAuxClass
cn:trCond
ibm-policyConditionName:trCond
ibm-idsConditionType:TR

dn:cn=RawPolicy, cn=attackRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRule
objectclass:ibm-policySubtreesPtrAuxClass
cn:RawPolicy
ibm-policyRuleName:RawPolicy
ibm-policyRuleEnabled:1
ibm-policyRuleConditionListType:2
ibm-policyRulePriority:255

dn:cn=actAssoc255, cn=RawPolicy, cn=attackRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleActionAssociation
objectclass:ibm-policyActionAuxClass
cn:actAssoc255
ibm-policyActionName:actAssoc255
```

```
ibm-policyActionOrder:0
ibm-policyActionDN:cn=RawAction, cn=attackActions, cn=policyRepository, ou=idsMgrPolicies,
o=ITSO,c=US

dn:cn=attack-type-255, cn=RawPolicy, cn=attackRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:attack-type-255
ibm-policyConditionName:attack-type-255
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:0
ibm-policyConditionDN:cn=attackOutRawCond, cn=condTypeCondRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US

dn:cn=attack-proto-255-0, cn=RawPolicy, cn=attackRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:attack-proto-255-0
ibm-policyConditionName:attack-proto-255-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:1
ibm-policyConditionDN:cn=Protocol_0, cn=ipProtocolRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US

dn:cn=ScanEventPolicy1, cn=scanEventRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRule
objectclass:ibm-policySubtreesPtrAuxClass
cn:ScanEventPolicy1
ibm-policyRuleName:ScanEventPolicy1
ibm-policyRuleEnabled:1
ibm-policyRuleConditionListType:2
ibm-policyRulePriority:255
ibm-policyKeyWords:Policy1

dn:cn=actAssoc255, cn=ScanEventPolicy1, cn=scanEventRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleActionAssociation
objectclass:ibm-policyActionAuxClass
cn:actAssoc255
ibm-policyActionName:actAssoc255
ibm-policyActionOrder:0
ibm-policyActionDN:cn=ScanEventAction1, cn=scanEventActions, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
ibm-policyKeyWords:Policy1

dn:cn=scanev-type-255, cn=ScanEventPolicy1, cn=scanEventRules, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:scanev-type-255
ibm-policyConditionName:scanev-type-255
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:0
ibm-policyConditionDN:cn=scanEventCond, cn=condTypeCondRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
ibm-policyKeyWords:Policy1

dn:cn=scanev-proto-255-0, cn=ScanEventPolicy1, cn=scanEventRules, ou=idsMgrPolicies,
o=ITSO,c=US
```

```
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:scanev-proto-255-0
ibm-policyConditionName:scanev-proto-255-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:1
ibm-policyConditionDN:cn=scan-tcp, cn=ipProtocolRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
ibm-policyKeyWords:Policy1

dn:cn=scanev-lport-255-0, cn=ScanEventPolicy1, cn=scanEventRules, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:scanev-lport-255-0
ibm-policyConditionName:scanev-lport-255-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:2
ibm-policyConditionDN:cn=Port_0, cn=scanLocalPortRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
ibm-policyKeyWords:Policy1

dn:cn=scanev-localAddr-255-0, cn=ScanEventPolicy1, cn=scanEventRules, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:scanev-localAddr-255-0
ibm-policyConditionName:scanev-localAddr-255-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:5
ibm-policyConditionDN:cn=IPAddr_0, cn=scanAddressRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
ibm-policyKeyWords:Policy1

dn:cn=ScanGlobalPolicy1, cn=scanGlobalRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRule
objectclass:ibm-policySubtreesPtrAuxClass
cn:ScanGlobalPolicy1
ibm-policyRuleName:ScanGlobalPolicy1
ibm-policyRuleEnabled:1
ibm-policyRuleConditionListType:2
ibm-policyRulePriority:255
ibm-policyKeyWords:Policy1

dn:cn=gScanassoc-255, cn=ScanGlobalPolicy1, cn=scanGlobalRules, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
objectclass:ibm-policyConditionAuxClass
objectclass:ibm-idsConditionAuxClass
objectclass:ibm-idsScanConditionAuxClass
cn:gScanassoc-255
ibm-policyConditionName:gScanassoc-255
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:0
ibm-idsConditionType:SCAN_GLOBAL
ibm-policyKeyWords:Policy1
```

```
dn:cn=gScanActAssoc255, cn=ScanGlobalPolicy1, cn=scanGlobalRules, ou=idsMgrPolicies,
o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleActionAssociation
objectclass:ibm-policyActionAuxClass
objectclass:ibm-idsActionAuxClass
objectclass:ibm-idsScanActionAuxClass
objectclass:ibm-idsNotificationAuxClass
cn:gScanActAssoc255
ibm-policyActionName:gScanActAssoc255
ibm-policyActionOrder:0
ibm-idsActionType:SCAN_GLOBAL
ibm-idsFSInterval:1
ibm-idsFSThreshold:5
ibm-idsSSInterval:120
ibm-idsSSThreshold:10
ibm-idsTypeActions:Statistics
ibm-idsTypeActions:EXCEPTSTATS
ibm-idsTypeActions:Log
ibm-idsTypeActions:Limit
ibm-idsNotification:SYSLOG
ibm-idsNotification:SYSLOGDETAIL
ibm-idsNotification:Console
ibm-idsStatInterval:60
ibm-idsLoggingLevel:0
ibm-idsTraceData:RECORDSIZE
ibm-idsTraceRecordSize:200
ibm-policyKeyWords:Policy1

dn:cn=TRPolicyTelnet, cn=trRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRule
objectclass:ibm-policySubtreesPtrAuxClass
cn:TRPolicyTelnet
ibm-policyRuleName:TRPolicyTelnet
ibm-policyRuleEnabled:1
ibm-policyRuleConditionListType:2
ibm-policyRulePriority:255
ibm-policyKeyWords:Policy1

dn:cn=actAssoc255, cn=TRPolicyTelnet, cn=trRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleActionAssociation
objectclass:ibm-policyActionAuxClass
cn:actAssoc255
ibm-policyActionName:actAssoc255
ibm-policyActionOrder:0
ibm-policyActionDN:cn=TRTCPAction1, cn=trTCPActions, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
ibm-policyKeyWords:Policy1

dn:cn=tr-type-255, cn=TRPolicyTelnet, cn=trRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:tr-type-255
ibm-policyConditionName:tr-type-255
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:0
ibm-policyConditionDN:cn=trCond, cn=condTypeCondRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
```

```
ibm-policyKeyWords:Policy1

dn:cn=tr-proto-255-0, cn=TRPolicyTelnet, cn=trRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:tr-proto-255-0
ibm-policyConditionName:tr-proto-255-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:1
ibm-policyConditionDN:cn=tr-tcp, cn=ipProtocolRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
ibm-policyKeyWords:Policy1

dn:cn=tr-lport-255-0, cn=TRPolicyTelnet, cn=trRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:tr-lport-255-0
ibm-policyConditionName:tr-lport-255-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:2
ibm-policyConditionDN:cn=Port_1, cn=trLocalPortRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
ibm-policyKeyWords:Policy1

dn:cn=tr-localAddr-255-0, cn=TRPolicyTelnet, cn=trRules, ou=idsMgrPolicies, o=ITSO,c=US
objectclass:ibm-policy
objectclass:ibm-policyRuleConditionAssociation
cn:tr-localAddr-255-0
ibm-policyConditionName:tr-localAddr-255-0
ibm-policyConditionNegated:FALSE
ibm-policyConditionGroupNumber:5
ibm-policyConditionDN:cn=IPAddr_3, cn=trAddressRepository, cn=policyRepository,
ou=idsMgrPolicies, o=ITSO,c=US
ibm-policyKeyWords:Policy1
```

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 300.

- *Communications Server for z/OS V1R2 TCP/IP Implementation Guide Volume 1: Base and TN3270 Configuration,* SG24-5227
- *Communications Server for z/OS V1R2 TCP/IP Implementation Guide Volume 4: Connectivity and Routing*, SG24-6516 (redpiece available at `http://www.ibm.com/redbooks`)
- *Communications Server for z/OS V1R2 TCP/IP Implementation Guide Volume 5: Availability, Scalability, and Performance*, SG24-6517 (redpiece available at `http://www.ibm.com/redbooks`)
- *Communications Server for z/OS V1R2 TCP/IP Implementation Guide Volume 7: Security*, SG24-6840 (redpiece available at `http://www.ibm.com/redbooks`)
- *IBM Communications Server for z/OS V2R10 TCP/IP Implementation Guide Volume 2: UNIX Applications,* SG24-5228
- *Managing OS/390 TCP/IP with SNMP*, SG24-5866
- *Migrating Subarea Networks to an IP Infrastructure Using Enterprise Extender*, SG24-5957
- *Networking with z/OS and Cisco Routers: An Interoperability Guide*, SG24-6297
- *OS/390 eNetwork Communications Server V2R7 TCP/IP Implementation Guide Volume 3: MVS Applications*, SG24-5229
- *Secure e-business in TCP/IP Networks on OS/390 and z/OS*, SG24-5383
- *TCP/IP in a Sysplex*, SG24-5235
- *TCP/IP Tutorial and Technical Overview*, GG24-3376
- *zSeries HiperSockets*, SG24-6816

Other resources

These publications are also relevant as further information sources:

- *z/OS V1R2.0 Security Server LDAP Server Administration and Use, SC24-5923*
- *z/OS V1R4.0 UNIX System Services User’s Guide*, SA22-7801
- *z/OS V1R2 Communications Server: IP Application Programming Interface Guide*, SC31-8788
- *z/OS V1R2 Communications Server: IP Configuration Guide*, SC31-8775
- *z/OS V1R2 Communications Server: IP Configuration Reference*, SC31-8776
- *z/OS V1R2 Communications Server: IP Messages Volume 1 (EZA)*, SC31-8783
- *z/OS V1R2 Communications Server: IP Messages Volume 2 (EZB)*, SC31-8784

- *z/OS V1R2 Communications Server: IP Messages Volume 3 (EZY)*, SC31-8785
- *z/OS V1R2 Communications Server: IP Messages Volume 4 (EZZ-SNM)*, SC31-8786
- *z/OS V1R2 Communications Server: IP Migration Guide*, GC31-8773
- *z/OS V1R2 Communications Server: IP System Administrator's Commands*, SC31-8781
- *z/OS V1R2 Communications Server: IP User's Guide and Commands*, SC31-8780
- *z/OS V1R3.0 C/C++ Programming Guide*, SC09-4765
- *z/OS V1R3.0 C/C++ Run-Time Library Reference*, SA22-7821
- *z/OS V1R3.0 MVS Initialization and Tuning Guide*, SA22-7591
- *z/OS V1R3.0 UNIX System Services Planning,* GA22-7800
- *z/OS V1R4 Communications Server: IP Diagnosis Guide*, GC31-8782
- *OS/390 V2R10.0 C/C++ Run-Time Library Reference,* SC28-1663
- *OS/390 V2R10.0 IBM Communications Server IP Configuration Guide,* SC-31-8725

Referenced Web sites

These Web sites are also relevant as further information sources:

- Download and installation instructions for Windows 2000 and Linux

  `http://www-3.ibm.com/software/network/commserver/downloads/zqosmanager.html`

- IBM Redbooks

  ibm.com/redbooks

- Java executable

  `http://java.sun.com/`

- NetSaint download

  `http://www.netsaint.org`

- Request for Comments (RFC) documents

  `http://www.ietf.org`

- z/OS Web pages

  `http://www-1.ibm.com/servers/eserver/zseries/zos/installation/installz12.html`

How to get IBM Redbooks

You can order hardcopy Redbooks, as well as view, download, or search for Redbooks at the following Web site:

ibm.com/redbooks

You can also download additional materials (code samples or diskette/CD-ROM images) from that site.

IBM Redbooks collections

Redbooks are also available on CD-ROMs. Click the CD-ROMs button on the Redbooks Web site for information about all the CD-ROMs offered, as well as updates and formats.

Index

Communications Server for z/OS V1R2 TCP/IP Implementation Guide Volume 6: Policy and Network Management

Communications Server for z/OS V1R2 TCP/IP Implementation Guide

Volume 6: Policy and Network Management

Redbooks

Design and implement z/OS policies in Policy Agent using zIDS and zQoS

Covers Sysplex Distributor, IDS, and QoS polices and samples

Details SNMP and strategies for network management

The Internet and enterprise-based networks have led to the rapidly increasing reliance upon TCP/IP implementations. The zSeries platform provides an environment upon which critical business applications flourish. The demands placed on these systems is ever-increasing and such demands require a solid, scalable, highly available, and highly performing Operating System and TCP/IP component. z/OS and Communications Server for z/OS provide for such a requirement with a TCP/IP stack that is robust and rich in functionality. The Communications Server for z/OS TCP/IP Implementation Guide series provides a comprehensive, in-depth survey of CS for z/OS.

Volume 6 covers the issues associated with managing the z/OS TCP/IP environment. First we describe the SNMP support in CS for z/OS IP and how SNMP can be used in conjunction with other elements in your network. Then we provide an in-depth look at managing, creating, and deploying policies in the z/OS Policy Agent. This includes policies with jurisdiction over Sysplex Distributor, Quality of Service (QoS), and Intrusion Detection Services (IDS).

Because of the varied scope of CS for z/OS, this redbook is not intended to cover all aspects of it. The main goal of this redbook is to provide insight into the different functionality available for managing the CS for z/OS environment efficiently through the use of SNMP and Policy Agent. For more information, including applications available with CS for z/OS IP, please reference the other Redbooks in the series.

SG24-6839-00 ISBN 0738426377